

---

# Indico Documentation

*Release 3.0.2*

**Indico Team**

**Sep 09, 2021**



---

## Contents

---

<b>1</b>	<b>Installation</b>	<b>3</b>
1.1	Installation guides . . . . .	3
<b>2</b>	<b>Configuration</b>	<b>47</b>
2.1	Configuration . . . . .	47
<b>3</b>	<b>Building</b>	<b>65</b>
3.1	Building . . . . .	65
<b>4</b>	<b>Search</b>	<b>67</b>
4.1	Search . . . . .	67
<b>5</b>	<b>Plugins</b>	<b>75</b>
5.1	Extending Indico with plugins . . . . .	75
<b>6</b>	<b>HTTP API</b>	<b>93</b>
6.1	Indico - HTTP API . . . . .	93
<b>7</b>	<b>API reference</b>	<b>121</b>
7.1	API reference . . . . .	121
<b>8</b>	<b>What's New</b>	<b>349</b>
8.1	Changelog . . . . .	349
<b>9</b>	<b>Indices and tables</b>	<b>377</b>
<b>10</b>	<b>Contact</b>	<b>379</b>
10.1	Contact . . . . .	379
	<b>Python Module Index</b>	<b>381</b>
	<b>Index</b>	<b>385</b>





*The effortless open source tool for event organization, archival and collaboration.*

license MIT pypi v3.0.2

Welcome to Indico's documentation. This documentation is split into several parts, from [installing Indico](#) to developing [Indico plugins](#). To dive into the internals of Indico, check out the [API documentation](#). Read more about Indico in our [official website](#).



To simply install and use Indico, follow the [production installation instructions](#). For those who are interested in developing new features and plugins for Indico, check out the [development installation instructions](#).

## 1.1 Installation guides

To simply install and use Indico, follow the [production installation instructions](#). For those who are interested in developing new features and plugins for Indico, check out the [development installation instructions](#).

### 1.1.1 Production

We provide guides to install Indico on CentOS and Debian systems. While other distributions are not officially supported, they should work fine, but the installation steps (especially package names) may need some slight adjustments.

Our guides cover a single-machine installation where Indico, Celery, Redis and PostgreSQL run on the same machine. This should be fine for almost all Indico instances, but adapting the steps to multiple machines is not particularly hard either.

#### CentOS / CC7

Except for minor differences, these guides apply to vanilla CentOS 7/8 and also the CERN flavor of CentOS 7, CC7 (CentOS CERN 7).

We have **not** tested the installation guides with CentOS Stream 8, as there are no up to date official Postgres packages available yet.

**Warning:** CentOS 8 is **only** supported with nginx, as some important packages (mod\_xsendfile and mod\_proxy\_uwsgi) are not (yet?) available for CentOS 8 in first-party repos. Once they are in EPEL, there is a good chance the guide will work as expected.

### nginx

---

**Note:** Please note that you **must** use Apache if you intend to use SSO using Shibboleth. If that's not the case because you do not use SSO at all or use e.g. OAuth, OIDC or SAML without Shibboleth, we recommend using nginx.

---

## 1. Enable EPEL

```
yum install -y epel-release
```

---

**Note:** If you use CC7, EPEL is already enabled and this step is not necessary

---

## 2. Install Packages

If you are on CentOS 7, edit `/etc/yum.repos.d/CentOS-Base.repo` and add `exclude=postgresql*` to the `[base]` and `[updates]` sections, as described in the [PostgreSQL wiki](#) and then run these commands:

```
yum install -y centos-release-scl
yum install -y https://download.postgresql.org/pub/repos/yum/repорpms/EL-7-x86_64/
↪pgdg-redhat-repo-latest.noarch.rpm
```

If you are on CentOS 8, run this instead:

```
dnf install -y https://download.postgresql.org/pub/repos/yum/repорpms/EL-8-x86_64/
↪pgdg-redhat-repo-latest.noarch.rpm
dnf -qy module disable postgresql
yum config-manager --set-enabled powertools
```

Now install all the required packages:

```
yum install -y postgresql13 postgresql13-server postgresql13-libs postgresql13-devel
↪postgresql13-contrib
yum install -y git gcc make redis nginx
yum install -y libjpeg-turbo-devel libxslt-devel libxml2-devel libffi-devel pcre-
↪devel libyaml-devel zlib-devel bzip2 bzip2-devel readline-devel sqlite sqlite-devel
↪openssl-devel xz xz-devel libffi-devel findutils libuuid-devel
/usr/pgsql-13/bin/postgresql-13-setup initdb
systemctl start postgresql-13.service redis.service
```

## 3. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;'
↪"
```



**Warning:** Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

## 4. Configure uWSGI & nginx

The default uWSGI and nginx configuration files should work fine in most cases.

```
cat > /etc/uwsgi-indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = nginx
umask = 027

processes = 4
enable-threads = true
chmod-socket = 770
socket = /opt/indico/web/uwsgi.sock
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF
```

We also need a systemd unit to start uWSGI.

```
cat > /etc/systemd/system/indico-uwsgi.service <<'EOF'
[Unit]
Description=Indico uWSGI
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/uwsgi --ini /etc/uwsgi-indico.ini
ExecReload=/bin/kill -HUP $MAINPID
Restart=always
SyslogIdentifier=indico-uwsgi
User=indico
Group=nginx
```

(continues on next page)

(continued from previous page)

```
UMask=0027
Type=notify
NotifyAccess=all
KillMode=mixed
KillSignal=SIGQUIT
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF
```

**Note:** Replace YOURHOSTNAME in the next file with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

```
cat > /etc/nginx/conf.d/indico.conf <<'EOF'
server {
    listen 80;
    listen [::]:80;
    server_name YOURHOSTNAME;
    return 301 https://$server_name$request_uri;
}

server {
    listen *:443 ssl http2;
    listen [::]:443 ssl http2 default ipv6only=on;
    server_name YOURHOSTNAME;

    ssl_certificate /etc/ssl/indico/indico.crt;
    ssl_certificate_key /etc/ssl/indico/indico.key;
    ssl_dhparam /etc/ssl/indico/ffdhe2048;

    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:10m;
    ssl_session_tickets off;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
→SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
→CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
→AES256-GCM-SHA384;
    ssl_prefer_server_ciphers off;

    access_log /opt/indico/log/nginx/access.log combined;
    error_log /opt/indico/log/nginx/error.log;

    if ($host != $server_name) {
        rewrite ^/(.*) https://$server_name/$1 permanent;
    }

    location /.xsf/indico/ {
        internal;
        alias /opt/indico/;
    }

    location ~ ^/(images|fonts)(.*)/(.+) (__v[0-9a-f]+)?\.(^[^.]*)$ {
```

(continues on next page)

(continued from previous page)

```

alias /opt/indico/web/static/$1$2/$3.$5;
access_log off;
}

location ~ ^/(css|dist|images|fonts)/(.*)$ {
    alias /opt/indico/web/static/$1/$2;
    access_log off;
}

location /robots.txt {
    alias /opt/indico/web/static/robots.txt;
    access_log off;
}

location / {
    root /var/empty/nginx;
    include /etc/nginx/uwsgi_params;
    uwsgi_pass unix:/opt/indico/web/uwsgi.sock;
    uwsgi_param UWSGI_SCHEME $scheme;
    uwsgi_read_timeout 15m;
    uwsgi_buffers 32 32k;
    uwsgi_busy_buffers_size 128k;
    uwsgi_hide_header X-Sendfile;
    client_max_body_size 1G;
}
}
EOF

```

## 5. Create a TLS Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```

mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico

```

We also use a strong set of pre-generated DH params (ffdhe2048 from RFC7919) as suggested in Mozilla's TLS config recommendations:

```

cat > /etc/ssl/indico/ffdhe2048 <<'EOF'
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEA/////////+t+FRYortKmq/cViAnPTzx2LnFg84tNpWp4TZBFGQz
+8yTnc4kmz75fS/jY2MMddj2gbICrsRhetPfHtXV/WVhJDP1H18GbtCFY2VVPe0a
87VXE15/V8k1mE8McODmi3fipona8+/och3xWKE2rec1MKzKT0g6eXq8CrGCsyT7
YdEIqUuyyOP7uWrat2DX9GgdT0Kj3j1N9K5W7edjcrsZCwenyO4KbXCeAvzhzffi
7MA0BM0oNC9hkXL+nOmFg/+OTxIy7vKBg8P+OxtMb61zO7X8vC7CIAXFjvGDfRaD
ssbzSibBsu/6iGtCOGEoXJf/////////wIBAg==
-----END DH PARAMETERS-----
EOF

```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

**Note:** Do not forget to replace `YOURHOSTNAME` with the same value you used above

```
openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt
```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

---

**Note:** There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the nginx config references a directory yet to be created, which prevents nginx from starting.

---

## 6. Configure SELinux

Indico works fine with SELinux enabled, but you need to load a custom SELinux module to tell SELinux about Indico's files and how they should be handled.

```
cat > /tmp/indico.cil <<'EOF'
; define custom type that logrotate can access
(type indico_log_t)
(typeattributeset file_type (indico_log_t))
(typeattributeset logfile (indico_log_t))
(roletype object_r indico_log_t)

; allow logrotate to reload systemd services
(allow logrotate_t init_t (service (start)))
(allow logrotate_t policykit_t (dbus (send_msg)))
(allow policykit_t logrotate_t (dbus (send_msg)))

; make sure the uwsgi socket is writable by the webserver
(typetransition unconfined_service_t usr_t sock_file "uwsgi.sock" httpd_sys_rw_
↪content_t)
(filecon "/opt/indico/web/uwsgi/.sock" socket (system_u object_r httpd_sys_rw_content_
↪t ((s0) (s0))))

; set proper types for our log dirs
(filecon "/opt/indico/log(/.*)?" any (system_u object_r indico_log_t ((s0) (s0))))
(filecon "/opt/indico/log/nginx(/.*)?" any (system_u object_r httpd_log_t ((s0) (s0))))
EOF
semodule -i /tmp/indico.cil
```

## 7. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
```

(continues on next page)

(continued from previous page)

```

Group=nginx
UMask=0027
Type=simple
KillMode=mixed
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF

systemctl daemon-reload

```

Now create a user that will be used to run Indico and switch to it:

```

useradd -rm -g nginx -d /opt/indico -s /bin/bash indico
su - indico

```

The first thing to do is installing pyenv - we use it to install the latest Python version as not all Linux distributions include it and like this Indico can benefit from the latest Python features.

```

curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash

cat >> ~/.bashrc <<'EOF'
export PATH="/opt/indico/.pyenv/bin:$PATH"
eval "$(pyenv init --path)"
eval "$(pyenv init -)"
EOF

source ~/.bashrc

```

You are now ready to install Python 3.9:

Run `pyenv install --list | egrep '^s*3\.9\.'` to check for the latest available version and then install it and set it as the active Python version (replace x in both lines).

```

pyenv install 3.9.x
pyenv global 3.9.x

```

This may take a while since pyenv needs to compile the specified Python version. Once done, you may want to use `python -V` to confirm that you are indeed using the version you just installed.

You are now ready to install Indico:

```

python -m venv --upgrade-deps --prompt indico ~/.venv
source ~/.venv/bin/activate
export PATH="/usr/pgsql-13/bin:$PATH"
echo 'source ~/.venv/bin/activate' >> ~/.bashrc
pip install wheel
pip install uwsgi
pip install indico

```

## 8. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when

asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/nginx
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/nginx
restorecon -R ~/
echo -e "\nSTATIC_FILE_METHOD = ('xaccelredirect', {'/opt/indico': '/.xsf/indico'})" >
↪> ~/etc/indico.conf
```

## 9. Create database schema

Finally you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

## 10. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart nginx.service indico-celery.service indico-uwsgi.service
systemctl enable nginx.service postgresql-13.service redis.service indico-celery.
↪service indico-uwsgi.service
```

## 11. Open the Firewall

```
firewall-cmd --permanent --add-port 443/tcp --add-port 80/tcp
firewall-cmd --reload
```

---

**Note:** This is only needed if you use CC7 as CentOS 7/8 have no firewall enabled by default

---

## 12. Optional: Get a Certificate from Let's Encrypt

To avoid ugly TLS warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
yum install -y python-certbot-nginx
certbot --nginx --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot-renew.timer
systemctl enable certbot-renew.timer
```

### 13. Create an Indico user

Access `https://YOURHOSTNAME` in your browser and follow the steps displayed there to create your initial user.

### 14. Install TeXLive

Follow the [LaTeX install guide](#) to install TeXLive so Indico can generate PDF files in various places.

## Apache

**Warning:** CentOS 8 with Apache is **not supported**, as some important packages (`mod_xsendfile` and `mod_proxy_uwsgi`) are not (yet?) available for CentOS 8 in first-party repos. Once they are in EPEL, there is a good chance this guide will work as expected.

#### 1. Enable EPEL

```
yum install -y epel-release
```

**Note:** If you use CC7, EPEL is already enabled and this step is not necessary

#### 2. Install Packages

If you are on CentOS 7, edit `/etc/yum.repos.d/CentOS-Base.repo` and add `exclude=postgresql*` to the `[base]` and `[updates]` sections, as described in the [PostgreSQL wiki](#) and then run these commands:

```
yum install -y centos-release-scl
yum install -y https://download.postgresql.org/pub/repos/yum/repорpms/EL-7-x86_64/
↳pgdg-redhat-repo-latest.noarch.rpm
```

If you are on CentOS 8, run this instead:

```
dnf install -y https://download.postgresql.org/pub/repos/yum/repорpms/EL-8-x86_64/
↳pgdg-redhat-repo-latest.noarch.rpm
dnf -qy module disable postgresql
yum config-manager --set-enabled powertools
```

```
yum install -y postgresql13 postgresql13-server postgresql13-libs postgresql13-devel
↳postgresql13-contrib
yum install -y git gcc make redis httpd mod_proxy_uwsgi mod_ssl mod_xsendfile
yum install -y libjpeg-turbo-devel libxslt-devel libxml2-devel libffi-devel pcre-
↳devel libyaml-devel zlib-devel bzip2 bzip2-devel readline-devel sqlite sqlite-devel
↳openssl-devel xz xz-devel libffi-devel findutils libuuid-devel
/usr/pgsql-13/bin/postgresql-13-setup initdb
systemctl start postgresql-13.service redis.service
```

### 3. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;
↵ "'
```

**Warning:** Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

### 4. Configure uWSGI & Apache

The default uWSGI and Apache configuration files should work fine in most cases.

```
cat > /etc/uwsgi-indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = apache
umask = 027

processes = 4
enable-threads = true
socket = 127.0.0.1:8008
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF
```

We also need a systemd unit to start uWSGI.

```
cat > /etc/systemd/system/indico-uwsgi.service <<'EOF'
[Unit]
```

(continues on next page)



(continued from previous page)

```

Description=Indico uWSGI
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/uwsgi --ini /etc/uwsgi-indico.ini
ExecReload=/bin/kill -HUP $MAINPID
Restart=always
SyslogIdentifier=indico-uwsgi
User=indico
Group=apache
UMask=0027
Type=notify
NotifyAccess=all
KillMode=mixed
KillSignal=SIGQUIT
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF

```

**Note:** Replace YOURHOSTNAME in the next files with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

```

cat > /etc/httpd/conf.d/indico-sslredirect.conf <<'EOF'
<VirtualHost *:80>
    ServerName YOURHOSTNAME
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>
EOF

cat > /etc/httpd/conf.d/indico.conf <<'EOF'
<VirtualHost *:443>
    ServerName YOURHOSTNAME
    DocumentRoot "/var/empty/apache"

    SSLEngine on
    SSLCertificateFile /etc/ssl/indico/indico.crt
    SSLCertificateChainFile /etc/ssl/indico/indico.crt
    SSLCertificateKeyFile /etc/ssl/indico/indico.key

    SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
    SSLCipherSuite ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
↪SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
↪CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384
    SSLHonorCipherOrder off
    SSLSessionTickets off

    XSendFile on
    XSendFilePath /opt/indico
    CustomLog /opt/indico/log/apache/access.log combined
    ErrorLog /opt/indico/log/apache/error.log

```

(continues on next page)

(continued from previous page)

```
LogLevel error
ServerSignature Off

<If "%{HTTP_HOST} != 'YOURHOSTNAME'">
    Redirect 301 / https://YOURHOSTNAME/
</If>

AliasMatch "^/(images|fonts) (.*)/(.+?) (__v[0-9a-f]+)?\.[^.]+" "/opt/indico/web/
↪static/$1$2/$3.$5"
AliasMatch "^/(css|dist|images|fonts) /(.*)$" "/opt/indico/web/static/$1/$2"
Alias /robots.txt /opt/indico/web/static/robots.txt

SetEnv UWSGI_SCHEME https
ProxyPass / uwsgi://127.0.0.1:8008/

<Directory /opt/indico>
    AllowOverride None
    Require all granted
</Directory>
</VirtualHost>
EOF
```

Now enable the uwsgi proxy module in apache:

```
echo 'LoadModule proxy_uwsgi_module modules/mod_proxy_uwsgi.so' > /etc/httpd/conf.
↪modules.d/proxy_uwsgi.conf
```

## 5. Create a TLS Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```
mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico
```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

---

**Note:** Do not forget to replace YOURHOSTNAME with the same value you used above

---

```
openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt
```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

---

**Note:** There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the Apache config references a directory yet to be created, which prevents Apache from starting.

---

## 6. Configure SELinux

Indico works fine with SELinux enabled, but you need to load a custom SELinux module to tell SELinux about Indico's files and how they should be handled.

```
cat > /tmp/indico.cil <<'EOF'
; define custom type that logrotate can access
(type indico_log_t)
(typeattributeset file_type (indico_log_t))
(typeattributeset logfile (indico_log_t))
(roletype object_r indico_log_t)

; allow logrotate to reload systemd services
(allow logrotate_t init_t (service (start)))
(allow logrotate_t policykit_t (dbus (send_msg)))
(allow policykit_t logrotate_t (dbus (send_msg)))

; make sure the uwsgi socket is writable by the webserver
(typetransition unconfined_service_t usr_t sock_file "uwsgi.sock" httpd_sys_rw_
↪content_t)
(filecon "/opt/indico/web/uwsgi\.sock" socket (system_u object_r httpd_sys_rw_content_
↪t ((s0) (s0))))

; set proper types for our log dirs
(filecon "/opt/indico/log(/.*)?" any (system_u object_r indico_log_t ((s0) (s0))))
(filecon "/opt/indico/log/apache(/.*)?" any (system_u object_r httpd_log_t_
↪((s0) (s0))))
EOF
semodule -i /tmp/indico.cil
```

## 7. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=apache
UMask=0027
Type=simple
KillMode=mixed
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF

systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g apache -d /opt/indico -s /bin/bash indico
su - indico
```

The first thing to do is installing pyenv - we use it to install the latest Python version as not all Linux distributions include it and like this Indico can benefit from the latest Python features.

```
curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash

cat >> ~/.bashrc <<'EOF'
export PATH="/opt/indico/.pyenv/bin:$PATH"
eval "$($pyenv init --path)"
eval "$($pyenv init -)"
EOF

source ~/.bashrc
```

You are now ready to install Python 3.9:

Run `pyenv install --list | egrep '^s*3\.9\.'` to check for the latest available version and then install it and set it as the active Python version (replace x in both lines).

```
pyenv install 3.9.x
pyenv global 3.9.x
```

This may take a while since pyenv needs to compile the specified Python version. Once done, you may want to use `python -V` to confirm that you are indeed using the version you just installed.

You are now ready to install Indico:

```
python -m venv --upgrade-deps --prompt indico ~/.venv
source ~/.venv/bin/activate
export PATH="$PATH:/usr/pgsql-13/bin"
echo 'source ~/.venv/bin/activate' >> ~/.bashrc
pip install wheel
pip install uwsgi
pip install indico
```

## 8. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/apache
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/apache
restorecon -R ~/
echo -e "\nSTATIC_FILE_METHOD = 'xsendfile'" >> ~/etc/indico.conf
```

## 9. Create database schema

Finally you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

## 10. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart httpd.service indico-celery.service indico-uwsgi.service
systemctl enable httpd.service postgresql-13.service redis.service indico-celery.
↪service indico-uwsgi.service
```

## 11. Open the Firewall

```
firewall-cmd --permanent --add-port 443/tcp --add-port 80/tcp
firewall-cmd --reload
```

---

**Note:** This is only needed if you use CC7 as CentOS 7/8 have no firewall enabled by default

---

## 12. Optional: Get a Certificate from Let's Encrypt

To avoid ugly TLS warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
yum install -y python-certbot-apache
certbot --apache --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot-renew.timer
systemctl enable certbot-renew.timer
```

## 13. Create an Indico user

Access `https://YOURHOSTNAME` in your browser and follow the steps displayed there to create your initial user.

## 14. Install TeXLive

Follow the [LaTeX install guide](#) to install TeXLive so Indico can generate PDF files in various places.

### Optional: Shibboleth

If your organization uses Shibboleth/SAML-based SSO, follow these steps to use it in Indico:

## 1. Install Shibboleth

Add the Shibboleth yum repository:

---

**Note:** If you use CC7, Shibboleth is already available and there is no need to add the repo manually.

---

If you use CentOS 7:

```
curl -fsSL -o /etc/yum.repos.d/shibboleth.repo 'https://shibboleth.net/cgi-bin/sp_
↪repo.cgi?platform=CentOS_7'
```

If you use CentOS 8:

```
curl -fsSL -o /etc/yum.repos.d/shibboleth.repo 'https://shibboleth.net/cgi-bin/sp_
↪repo.cgi?platform=CentOS_8'
```

Now install Shibboleth itself. When prompted to accept the GPG key of the Shibboleth yum repo, confirm the prompt.

```
setsebool httpd_can_network_connect 1
yum install -y shibboleth xmltooling-schemas opensaml-schemas
```

## 2. Configure Shibboleth

This is outside the scope of this documentation and depends on your environment (Shibboleth, SAML, ADFS, etc). Please contact whoever runs your SSO infrastructure if you need assistance.

## 3. Enable Shibboleth in Apache

Add the following code to your `/etc/httpd/conf.d/indico.conf` right before the `AliasMatch` lines:

```
<LocationMatch "^(/Shibboleth\.sso|/login/shib-sso/shibboleth)">
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  ShibExportAssertion Off
  Require valid-user
</LocationMatch>
```

## 4. Enable Shibboleth in Indico

Add the following code to your `/opt/indico/etc/indico.conf`:

```
# SSO
AUTH_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'attrs_prefix': 'ADFS_',
        'callback_uri': '/login/shib-sso/shibboleth',
        # 'logout_uri': 'https://login.yourcompany.tld/logout'
    }
}
```

(continues on next page)

(continued from previous page)

```

IDENTITY_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'identifier_field': 'ADFS_LOGIN',
        'mapping': {
            'affiliation': 'ADFS_HOMEINSTITUTE',
            'first_name': 'ADFS_FIRSTNAME',
            'last_name': 'ADFS_LASTNAME',
            'email': 'ADFS_EMAIL',
            'phone': 'ADFS_PHONENUMBER'
        },
        'trusted_email': True
    }
}

```

The values for `attrs_prefix`, `mapping` and `identifier_field` may be different in your environment. Uncomment and set `logout_uri` if your SSO infrastructure provides a logout URL (usually used to log you out from all applications).

If you only want to use SSO, without allowing people to login locally using username/password, disable it by setting `LOCAL_IDENTITY_PROVIDERS = False` in `indico.conf`.

**Warning:** We assume that emails received from SSO are already validated. If this is not the case, make sure to disable `trusted_email` which will require email validation in Indico when logging in for the first time. Otherwise people could take over the account of someone else by using their email address!

**Note:** The example config is rather simple and only accesses data from SSO during login. This is not sufficient for advanced features such as automatic synchronization of names, affiliations and phone numbers or using centrally managed groups. To use these features, you need to use e.g. the LDAP identity provider and use the information received via SSO to retrieve the user details from LDAP. If you need assistance with this, feel free to ask us on IRC ([#indico](#) @ Libera.Chat) or [the forum](#).

**Note:** Please note that you **must** use Apache if you intend to use SSO using Shibboleth. If that's not the case because you do not use SSO at all or use e.g. OAuth, OIDC or SAML without Shibboleth, we recommend using nginx.

## Debian / Ubuntu

Except for minor differences, this guide applies to both Debian and Ubuntu. It has been tested with Debian 10 (Buster) and Ubuntu 20.04 (Focal).

**Warning:** Older distributions may work, but they are unsupported. We do not recommend using those unless you have a strong reason for it and the necessary system administration knowledge to know how to deal with compatibility issues should any arise.

## nginx

**Note:** Please note that you **must** use Apache if you intend to use SSO using Shibboleth. If that's not the case because you do not use SSO at all or use e.g. OAuth, OIDC or SAML without Shibboleth, we recommend using nginx.

---

## 1. Install Packages

PostgreSQL and nginx are installed from their upstream repos to get much more recent versions.

```
apt install -y lsb-release wget curl gnupg
echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main" > /
↳etc/apt/sources.list.d/pgdg.list
echo "deb http://nginx.org/packages/$(lsb_release -is | tr '[:upper:]' '[:lower:]')/
↳$(lsb_release -cs) nginx" > /etc/apt/sources.list.d/nginx.list
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | apt-key add -
wget --quiet -O - https://nginx.org/keys/nginx_signing.key | apt-key add -
apt update
apt install -y --install-recommends postgresql-13 libpq-dev nginx libxslt1-dev
↳libxml2-dev libffi-dev libpcres3-dev libyaml-dev libssl-dev zlib1g-dev libbz2-dev
↳libreadline-dev libsqlite3-dev libncurses5-dev libncursesw5-dev xz-utils libffi-dev
↳liblzma-dev uuid-dev build-essential redis-server
```

If you use Debian, run this command:

```
apt install -y libjpeg62-turbo-dev
```

If you use Ubuntu, run this instead:

```
apt install -y libjpeg-turbo8-dev zlib1g-dev
```

Afterwards, make sure the services you just installed are running:

```
systemctl start postgresql.service redis-server.service
```

## 2. Create a Database

Let's create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser).

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;
↳"'
```

**Warning:** Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

## 3. Configure uWSGI & nginx

The default uWSGI and nginx configuration files should work fine in most cases.



```

cat > /etc/uwsgi-indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = nginx
umask = 027

processes = 4
enable-threads = true
chmod-socket = 770
chown-socket = indico:nginx
socket = /opt/indico/web/uwsgi.sock
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF

```

We also need a systemd unit to start uWSGI.

```

cat > /etc/systemd/system/indico-uwsgi.service <<'EOF'
[Unit]
Description=Indico uWSGI
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/uwsgi --ini /etc/uwsgi-indico.ini
ExecReload=/bin/kill -HUP $MAINPID
Restart=always
SyslogIdentifier=indico-uwsgi
User=indico
Group=nginx
UMask=0027
Type=notify
NotifyAccess=all
KillMode=mixed
KillSignal=SIGQUIT
TimeoutStopSec=300

[Install]

```

(continues on next page)

(continued from previous page)

```
WantedBy=multi-user.target
EOF
```

**Note:** Replace YOURHOSTNAME in the next file with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

```
cat > /etc/nginx/conf.d/indico.conf <<'EOF'
server {
    listen 80;
    listen [::]:80;
    server_name YOURHOSTNAME;
    return 301 https://$server_name$request_uri;
}

server {
    listen      *:443 ssl http2;
    listen      [::]:443 ssl http2 default ipv6only=on;
    server_name YOURHOSTNAME;

    ssl_certificate      /etc/ssl/indico/indico.crt;
    ssl_certificate_key  /etc/ssl/indico/indico.key;
    ssl_dhparam          /etc/ssl/indico/ffdhe2048;

    ssl_session_timeout  1d;
    ssl_session_cache    shared:SSL:10m;
    ssl_session_tickets  off;
    ssl_protocols        TLSv1.2 TLSv1.3;
    ssl_ciphers           ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
↪SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
↪CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384;
    ssl_prefer_server_ciphers off;

    access_log            /opt/indico/log/nginx/access.log combined;
    error_log             /opt/indico/log/nginx/error.log;

    if ($host != $server_name) {
        rewrite ^/(.*) https://$server_name/$1 permanent;
    }

    location /.xsf/indico/ {
        internal;
        alias /opt/indico/;
    }

    location ~ ^/(images|fonts)(.*)/(.+) (__v[0-9a-f]+)?\.[^.]$ {
        alias /opt/indico/web/static/$1$2/$3.$5;
        access_log off;
    }

    location ~ ^/(css|dist|images|fonts)/(.*)$ {
        alias /opt/indico/web/static/$1/$2;
        access_log off;
    }
}
```

(continues on next page)

(continued from previous page)

```

location /robots.txt {
    alias /opt/indico/web/static/robots.txt;
    access_log off;
}

location / {
    root /var/empty/nginx;
    include /etc/nginx/uwsgi_params;
    uwsgi_pass unix:/opt/indico/web/uwsgi.sock;
    uwsgi_param UWSGI_SCHEME $scheme;
    uwsgi_read_timeout 15m;
    uwsgi_buffers 32 32k;
    uwsgi_busy_buffers_size 128k;
    uwsgi_hide_header X-Sendfile;
    client_max_body_size 1G;
}
}
EOF

```

#### 4. Create a TLS Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```

mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico

```

We also use a strong set of pre-generated DH params (ffdhe2048 from RFC7919) as suggested in Mozilla's TLS config recommendations:

```

cat > /etc/ssl/indico/ffdhe2048 <<'EOF'
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEA////////+t+FRYortKmq/cViAnPTzx2LnFg84tNpWp4TZBFGQz
+8yTnc4kmz75fS/jY2MMddj2gbICrsRhetPfHtXV/WVhJDP1H18GbtCFY2VVPe0a
87VXE15/V8k1mE8McODmi3fipona8+/och3xWKE2rec1MKzKT0g6eXq8CrGCsyT7
YdEIqUuyyOP7uWrat2DX9GgdT0Kj3j1N9K5W7edjcrsZCwenyO4KbXCeAvzhzffi
7MA0BM0oNC9hkXL+nOmFg/+OTxIy7vKBg8P+OxtMb61zO7X8vC7CIAXFjvGDfRaD
ssbzSibBsu/6iGtCOGEoXJf////////wIBAg==
-----END DH PARAMETERS-----
EOF

```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

---

**Note:** Do not forget to replace YOURHOSTNAME with the same value you used above

---

```

openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt

```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

**Note:** There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the nginx config references a directory yet to be created, which prevents nginx from starting.

---

## 5. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=nginx
UMask=0027
Type=simple
KillMode=mixed
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF

systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g nginx -d /opt/indico -s /bin/bash indico
su - indico
```

The first thing to do is installing pyenv - we use it to install the latest Python version as not all Linux distributions include it and like this Indico can benefit from the latest Python features.

```
curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash

cat >> ~/.bashrc <<'EOF'
export PATH="/opt/indico/.pyenv/bin:$PATH"
eval "$(pyenv init --path)"
eval "$(pyenv init -)"
EOF

source ~/.bashrc
```

You are now ready to install Python 3.9:

Run `pyenv install --list | egrep '^s*3\.9\.'` to check for the latest available version and then install it and set it as the active Python version (replace x in both lines).

```
pyenv install 3.9.x
pyenv global 3.9.x
```

This may take a while since pyenv needs to compile the specified Python version. Once done, you may want to use `python -V` to confirm that you are indeed using the version you just installed.

You are now ready to install Indico:

```
python -m venv --upgrade-deps --prompt indico ~/.venv
source ~/.venv/bin/activate
echo 'source ~/.venv/bin/activate' >> ~/.bashrc
pip install wheel
pip install uwsgi
pip install indico
```

## 6. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/nginx
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/nginx
echo -e "\nSTATIC_FILE_METHOD = ('xaccelredirect', {'/opt/indico': '/.xsf/indico'})" >
↪> ~/etc/indico.conf
```

## 7. Create database schema

Finally, you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

## 8. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart nginx.service indico-celery.service indico-uwsgi.service
systemctl enable nginx.service postgresql.service redis-server.service indico-celery.
↪service indico-uwsgi.service
```

## 9. Optional: Get a Certificate from Let's Encrypt

**Note:** You need to use at least Debian 9 (Stretch) to use certbot. If you are still using Debian 8 (Jessie), consider updating or install certbot from backports.

If you use Ubuntu, install the certbot PPA:

```
apt install -y software-properties-common
add-apt-repository -y ppa:certbot/certbot
apt update
```

To avoid ugly TLS warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
apt install -y python-certbot-nginx
certbot --nginx --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot.timer
systemctl enable certbot.timer
```

## 10. Create an Indico user

Access <https://YOURHOSTNAME> in your browser and follow the steps displayed there to create your initial user.

## 11. Install TeXLive

Follow the [LaTeX install guide](#) to install TeXLive so Indico can generate PDF files in various places.

## Apache

### 1. Install Packages

PostgreSQL is installed from its upstream repos to get a much more recent version.

```
apt install -y lsb-release wget curl gnupg
echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main" > /
↳etc/apt/sources.list.d/pgdg.list
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | apt-key add -
apt update
apt install -y --install-recommends postgresql-13 libpq-dev apache2 libapache2-mod-
↳proxy-uwsgi libapache2-mod-xsendfile libxslt1-dev libxml2-dev libffi-dev libpcre3-
↳dev libyaml-dev libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev
↳libncurses5-dev libncursesw5-dev xz-utils libffi-dev liblzma-dev uuid-dev build-
↳essential redis-server
```

If you use Debian, run this command:

```
apt install -y libjpeg62-turbo-dev
```

If you use Ubuntu, run this instead:

```
apt install -y libjpeg-turbo8-dev zlib1g-dev
```

Afterwards, make sure the services you just installed are running:

```
systemctl start postgresql.service redis-server.service
```

## 2. Create a Database

Let's create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser).

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;"
↵'
```

**Warning:** Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

## 3. Configure uWSGI & Apache

The default uWSGI and Apache configuration files should work fine in most cases.

```
cat > /etc/uwsgi-indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = www-data
umask = 027

processes = 4
enable-threads = true
socket = 127.0.0.1:8008
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF
```

We also need a systemd unit to start uWSGI.

```
cat > /etc/systemd/system/indico-uwsgi.service <<'EOF'
[Unit]
```

(continues on next page)

(continued from previous page)

```
Description=Indico uWSGI
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/uwsgi --ini /etc/uwsgi-indico.ini
ExecReload=/bin/kill -HUP $MAINPID
Restart=always
SyslogIdentifier=indico-uwsgi
User=indico
Group=www-data
UMask=0027
Type=notify
NotifyAccess=all
KillMode=mixed
KillSignal=SIGQUIT
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF
```

---

**Note:** Replace YOURHOSTNAME in the next files with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

---

```
cat > /etc/apache2/sites-available/indico-sslredirect.conf <<'EOF'
<VirtualHost *:80>
    ServerName YOURHOSTNAME
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>
EOF

cat > /etc/apache2/sites-available/indico.conf <<'EOF'
<VirtualHost *:443>
    ServerName YOURHOSTNAME
    DocumentRoot "/var/empty/apache"
    Protocols h2 http/1.1

    SSLEngine on
    SSLCertificateFile /etc/ssl/indico/indico.crt
    SSLCertificateKeyFile /etc/ssl/indico/indico.key

    SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
    SSLCipherSuite ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
↪SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
↪CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384
    SSLHonorCipherOrder off
    SSLSessionTickets off

    XSendFile on
    XSendFilePath /opt/indico
    CustomLog /opt/indico/log/apache/access.log combined
    ErrorLog /opt/indico/log/apache/error.log
```

(continues on next page)



(continued from previous page)

```

LogLevel error
ServerSignature Off

<If "%{HTTP_HOST} != 'YOURHOSTNAME'">
    Redirect 301 / https://YOURHOSTNAME/
</If>

AliasMatch "^/(images|fonts) (.*)/(.+?) (__v[0-9a-f]+)?\.[^.]+" "/opt/indico/web/
→static/$1$2/$3.$5"
AliasMatch "^/(css|dist|images|fonts) /(.*)$" "/opt/indico/web/static/$1/$2"
Alias /robots.txt /opt/indico/web/static/robots.txt

SetEnv UWSGI_SCHEME https
ProxyPass / uwsgi://127.0.0.1:8008/

<Directory /opt/indico>
    AllowOverride None
    Require all granted
</Directory>
</VirtualHost>
EOF

```

Now enable the necessary modules and the indico site in apache:

```

a2enmod proxy_uwsgi rewrite ssl xsendfile
a2dissite 000-default
a2ensite indico indico-sslredir

```

#### 4. Create a TLS Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```

mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico

```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

---

**Note:** Do not forget to replace YOURHOSTNAME with the same value you used above

---

```

openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
→indico/indico.key -out /etc/ssl/indico/indico.crt

```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

---

**Note:** There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the Apache config references a directory yet to be created, which prevents Apache from starting.

---

## 5. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=www-data
UMask=0027
Type=simple
KillMode=mixed
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
EOF

systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g www-data -d /opt/indico -s /bin/bash indico
su - indico
```

The first thing to do is installing pyenv - we use it to install the latest Python version as not all Linux distributions include it and like this Indico can benefit from the latest Python features.

```
curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash

cat >> ~/.bashrc <<'EOF'
export PATH="/opt/indico/.pyenv/bin:$PATH"
eval "$(pyenv init --path)"
eval "$(pyenv init -)"
EOF

source ~/.bashrc
```

You are now ready to install Python 3.9:

Run `pyenv install --list | egrep '^s*3\.9\.'` to check for the latest available version and then install it and set it as the active Python version (replace `x` in both lines).

```
pyenv install 3.9.x
pyenv global 3.9.x
```

This may take a while since pyenv needs to compile the specified Python version. Once done, you may want to use `python -V` to confirm that you are indeed using the version you just installed.

You are now ready to install Indico:

```
python -m venv --upgrade-deps --prompt indico ~/.venv
source ~/.venv/bin/activate
echo 'source ~/.venv/bin/activate' >> ~/.bashrc
pip install wheel
pip install uwsgi
pip install indico
```

## 6. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/apache
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/apache
echo -e "\nSTATIC_FILE_METHOD = 'xsendfile'" >> ~/etc/indico.conf
```

## 7. Create database schema

Finally, you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

## 8. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart apache2.service indico-celery.service indico-uwsgi.service
systemctl enable apache2.service postgresql.service redis-server.service indico-
celery.service indico-uwsgi.service
```

## 9. Optional: Get a Certificate from Let's Encrypt

**Note:** You need to use at least Debian 9 (Stretch) to use certbot. If you are still using Debian 8 (Jessie), consider updating or install certbot from backports.

If you use Ubuntu, install the certbot PPA:

```
apt install -y software-properties-common
add-apt-repository -y ppa:certbot/certbot
apt update
```

To avoid ugly TLS warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
apt install -y python-certbot-apache
certbot --apache --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot.timer
systemctl enable certbot.timer
```

## 10. Create an Indico user

Access `https://YOURHOSTNAME` in your browser and follow the steps displayed there to create your initial user.

## 11. Install TeXLive

Follow the [LaTeX install guide](#) to install TeXLive so Indico can generate PDF files in various places.

### Optional: Shibboleth

If your organization uses Shibboleth/SAML-based SSO, follow these steps to use it in Indico:

#### 1. Install Shibboleth

```
apt install -y libapache2-mod-shib2
a2enmod shib2
```

#### 2. Configure Shibboleth

This is outside the scope of this documentation and depends on your environment (Shibboleth, SAML, ADFS, etc). Please contact whoever runs your SSO infrastructure if you need assistance.

#### 3. Enable Shibboleth in Apache

Add the following code to your `/etc/apache2/sites-available/indico.conf` right before the `AliasMatch` lines:

```
<LocationMatch "^(/Shibboleth\.sso|/login/shib-sso/shibboleth)">
    AuthType shibboleth
    ShibRequestSetting requireSession 1
    ShibExportAssertion Off
    Require valid-user
</LocationMatch>
```

## 4. Enable Shibboleth in Indico

Add the following code to your `/opt/indico/etc/indico.conf`:

```
# SSO
AUTH_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'attrs_prefix': 'ADFS_',
        'callback_uri': '/login/shib-sso/shibboleth',
        # 'logout_uri': 'https://login.yourcompany.tld/logout'
    }
}
IDENTITY_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'identifier_field': 'ADFS_LOGIN',
        'mapping': {
            'affiliation': 'ADFS_HOMEINSTITUTE',
            'first_name': 'ADFS_FIRSTNAME',
            'last_name': 'ADFS_LASTNAME',
            'email': 'ADFS_EMAIL',
            'phone': 'ADFS_PHONENUMBER'
        },
        'trusted_email': True
    }
}
```

The values for `attrs_prefix`, `mapping` and `identifier_field` may be different in your environment. Uncomment and set `logout_uri` if your SSO infrastructure provides a logout URL (usually used to log you out from all applications).

If you only want to use SSO, without allowing people to login locally using username/password, disable it by setting `LOCAL_IDENTITY_PROVIDERS = False` in `indico.conf`.

**Warning:** We assume that emails received from SSO are already validated. If this is not the case, make sure to disable `trusted_email` which will require email validation in Indico when logging in for the first time. Otherwise people could take over the account of someone else by using their email address!

**Note:** The example config is rather simple and only accesses data from SSO during login. This is not sufficient for advanced features such as automatic synchronization of names, affiliations and phone numbers or using centrally managed groups. To use these features, you need to use e.g. the LDAP identity provider and use the information received via SSO to retrieve the user details from LDAP. If you need assistance with this, feel free to ask us on IRC ([#indico @ Libera.Chat](#)) or [the forum](#).

**Note:** Please note that you **must** use Apache if you intend to use SSO using Shibboleth. If that's not the case because you do not use SSO at all or use e.g. OAuth, OIDC or SAML without Shibboleth, we recommend using nginx.

### 1.1.2 Upgrade

It is important to keep your Indico instance up to date to have the latest bug fixes and features. Upgrading can be done with almost no user-facing downtime.

**Warning:** When upgrading a production system it is highly recommended to create a database backup before starting.

#### Upgrading between 3.x versions

First of all, stop the Celery worker. To do so, run this as *root*:

```
systemctl stop indico-celery.service
```

Now switch to the *indico* user and activate the virtualenv:

```
su - indico
source ~/.venv/bin/activate
```

If you are on CentOS, update your PATH to avoid errors in case the new Indico version needs to install an updated version of the PostgreSQL client library (psycopg2):

```
export PATH="$PATH:/usr/pgsql-13/bin"
```

You are now ready to install the latest version of Indico:

```
pip install -U indico
```

If you installed the official plugins, update them too:

```
pip install -U indico-plugins
```

It is a good idea to ensure you are using the latest recommended Python version:

```
indico setup upgrade-python
```

Some versions may include database schema upgrades. Make sure to perform them immediately after upgrading. If there are no schema changes, the command will simply do nothing.

```
indico db upgrade
indico db --all-plugins upgrade
```

**Note:** Some database structure changes require an *exclusive lock* on some tables in the database. Unless you have very high activity on your instance, this lock can be acquired quickly, but if the upgrade command seems to hang for more than a few seconds, you can restart uWSGI by running `systemctl restart uwsgi.service` as *root* (in a separate shell, i.e. don't abort the upgrade command!) which will ensure nothing is accessing Indico for a moment.

Unless you just restarted uWSGI, it is now time to reload it so the new version is actually used:

```
touch ~/web/indico.wsgi
```

Also start the Celery worker again (once again, as *root*):

```
systemctl start indico-celery.service
```

## Upgrading from 2.x to 3.x

The upgrade from 2.x to 3.x is a major change since Indico now requires Python 3. We also strongly recommend upgrading your database to PostgreSQL 13.

**Note:** There are no changes that require the newer Postgres version immediately, but we no longer test on versions older than Postgres 12, and thus can give you no guarantees that things will keep working on older versions such as 9.6.

**Warning:** If you are using any custom plugins they will most likely no longer work and need to be updated. Contact the developers of these plugins to see whether they already have a version compatible with Python 3 and Indico 3.

Due to the significant changes in the environment, we recommend using a **freshly installed server/VM** with the latest long-term-supported version of your preferred Linux distribution.

**Note:** If you are using CentOS, staying with CentOS 7 is recommended as CentOS 8 actually has a much earlier end-of-life date (end of 2021) than CentOS 7 (mid 2024), and running Indico with Apache on CentOS 8 is currently not supported.

When following the *production installation guide*, there are a few places where you need to do something differently:

- Instead of running `indico db prepare`, restore a dump of your old Postgres database
- You still need to run `indico setup wizard` to create some of the directories, but compare the generated config file with your old one and update any settings you may have changed manually (e.g. for LDAP or SSO authentication)
- Copy the contents of the `/opt/indico/archive` folder from your old instance and ensure owner, group and permissions are correct. This step is critical as this folder contains all the files uploaded to Indico

If you need any help with the upgrade or encounter any issues, please open a thread in [our forum](#).

## Upgrading from 2.x to 3.x in-place

**Warning:** If you are not experienced with Linux system administration, we highly recommend you to either ask someone from your IT department for assistance and/or follow our recommendation of using a new server/VM to install Indico v3.

In case you prefer to perform the upgrade in place on your existing server, you will need to compare the installation guides of 2.3 and 3.x and apply the differences manually. This should be fairly easy for someone with Linux system administration experience, but here are some important points:

- Create a backup of both your Postgres database and `/opt/indico/archive`
- Stop, disable and and uninstall uWSGI and delete the old config file. To support the latest Python version uWSGI is now installed into the Indico virtual environment using `pip`

- Delete the `~/ .venv` folder of the Indico user and recreate it using the commands from the setup guide
- Make sure to update your webserver config to use the more modern TLS defaults

### 1.1.3 Upgrade Indico from 1.2

The migration tool ([indico-migrate](#)) requires Python 2.7 and Indico 2.0. It is not supported by Indico v3 nor will it work on Python 3.

If you still need to migrate a legacy instance from the 1.x (or older), please consult the [documentation from Indico v2](#). You may also want to consider running the migration on a separate virtual machine in order to not clutter the server that will run Indico v3 with legacy tools and software.

### 1.1.4 Installation guide (development)

#### Installing System Packages

Web assets such as JavaScript and SCSS files are compiled using [Webpack](#), which requires NodeJS to be present. You can find information on how to install NodeJS [here](#).

Do not use the default NodeJS packages from your Linux distribution as they are usually outdated or come with an outdated npm version.

Since only few Linux distributions include Python 3.9 in their package managers, we recommend installing [pyenv](#) and then install the latest Python 3.9 version using `pyenv install 3.9.6` (adapt this command in case a newer version is available).

---

**Tip:** You can run `pyenv doctor` once you installed and enabled pyenv in order to see whether all dependencies are met. There's a good chance that you need to install some additional system packages beyond those listed below, and using this tool will tell you what exactly you need.

---

#### CentOS/Fedora

```
yum install -y gcc redis libjpeg-turbo-devel libxslt-devel libxml2-devel \
    libffi-devel pcre-devel libyaml-devel redhat-rpm-config \
    postgresql postgresql-server postgresql-contrib libpq-devel
systemctl start redis.service postgresql.service
```

#### Debian/Ubuntu

```
apt install -y --install-recommends libxslt1-dev libxml2-dev libffi-dev libpcre3-dev \
    libyaml-dev build-essential redis-server postgresql libpq-dev
```

Then on Debian:

```
apt install -y libjpeg62-turbo-dev
```

And on Ubuntu:



```
apt install -y libjpeg-turbo8-dev zlib1g-dev
```

## macOS

We recommend that you use [Homebrew](#):

```
brew install redis libjpeg libffi pcre libyaml postgresql
brew services start postgresql
brew services start redis
```

## Creating the directory structure

You will need a directory in your file system to store Indico as well as its data files (archives, etc. . .). Some developers keep all their code inside a `dev` or `code` dir. We will assume `dev` here.

```
mkdir -p ~/dev/indico/data
```

We will need a `virtualenv` where to run Indico:

```
cd ~/dev/indico
pyenv local 3.9.6
python -m venv env
```

**Note:** After setting the version with `pyenv`, it's a good idea to use `python -V` to ensure you are really running that particular Python version; depending on the shell you may need to restart your shell first. In case you installed a newer version than 3.9.6 earlier, adapt the `pyenv` command accordingly.

## Cloning Indico

First, let's clone Indico's code base. If you're going to contribute back to the project, it's probably best if you clone your own [GitHub fork of the project](#) and set it as the origin:

```
git clone git@github.com:<your-github-username>/indico.git src
cd src
git remote add upstream https://github.com/indico/indico.git
cd ..
```

Otherwise, cloning the upstream repository as the origin should be enough:

```
git clone https://github.com/indico/indico.git src
```

If you're going to be changing the standard Indico plugins and/or the documentation, you can also clone those:

```
mkdir plugins
git clone https://github.com/indico/indico-plugins.git plugins/base
git clone https://github.com/indico/indico-user-docs.git user-docs
```

## Setting up Mailedump (recommended)

Some actions in Indico trigger automatic e-mails. Those will normally have to be routed through an SMTP server. This can become a problem if you're using production data and/or real e-mails, as users may end up being spammed unnecessarily. This is why we advise that you include a fake SMTP server in your development setup. **Mailedump** does exactly this and runs on Python. It should be quite simple to set up:

```
python -m venv maileddump
./maileddump/bin/pip install -U pip setuptools wheel
./maileddump/bin/pip install maileddump
./maileddump/bin/maileddump -p /tmp/maileddump.pid
```

You'll then be able to access the message log at <http://localhost:1080>.

## Creating the DB

```
sudo -u postgres createuser $USER --createdb
sudo -u postgres createdb indico_template -O $USER
sudo -u postgres psql indico_template -c "CREATE EXTENSION unaccent; CREATE EXTENSION_
↳ pg_trgm;"
createdb indico -T indico_template
```

## Configuring

Let's get into the Indico virtualenv:

```
source ./env/bin/activate
pip install -U pip setuptools wheel

cd src
pip install -e '.[dev]'
npm ci
```

Then, follow the instructions given by the wizard:

```
indico setup wizard --dev
```

You can then initialize the DB:

```
indico db prepare
```

To build the locales, use:

```
indico i18n compile-catalog
indico i18n compile-catalog-react
```

## Running Indico

You will need two shells running in parallel. The first one will run the webpack watcher, which compiles the JavaScript and style assets every time you change them:

```
./bin/maintenance/build-assets.py indico --dev --watch
```

On the second one we'll run the Indico Development server:

```
indico run -h <your-hostname> -q --enable-evalcx
```

Double-check that your hostname matches that which has been set in the config file (by the wizard).

It is also worth mentioning that when working on a plugin, it is necessary to run another webpack watcher to build the plugin assets. That can be accomplished using the same command as above with an argument specifying which plugin you want to build the assets for:

```
./bin/maintenance/build-assets.py <plugin-name> --dev --watch
```

You can also build the assets for all the plugins:

```
./bin/maintenance/build-assets.py all-plugins --dev <plugins-directory>
```

## Installing TeXLive (optional)

If you need PDF generation in certain parts of Indico to work (e.g. for contributions and the Book of Abstracts), you need LaTeX. To install it, follow the [LaTeX install guide](#).

## Using HTTPS through nginx (optional)

If you wish to open your development server to others, then we highly recommend that you properly set HTTPS. While you could do so directly at the development server, it's normally easier to proxy it through nginx and have it serve static files as well.

You should obviously install nginx first:

```
sudo yum install nginx # centos/fedora
sudo apt install nginx # debian/ubuntu
brew install nginx     # macOS
```

Here is an example of a `nginx.conf` you can use. It assumes your username is `jdope` and the hostname is `acme.example.org`:

```
user jdope users;
worker_processes 4;
error_log /var/log/nginx/error.log info;
pid /run/nginx.pid;

events {
    worker_connections 1024;
    use epoll;
}

http {
    access_log off;

    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;

    keepalive_timeout 75 20;
    types_hash_max_size 2048;
```

(continues on next page)

(continued from previous page)

```

ignore_invalid_headers on;

connection_pool_size 256;
client_header_buffer_size 10k;
large_client_header_buffers 4 20k;
request_pool_size 4k;
client_max_body_size 2048m;

proxy_buffers 32 32k;
proxy_buffer_size 32k;
proxy_busy_buffers_size 128k;

gzip on;
gzip_min_length 1100;
gzip_buffers 4 8k;
gzip_types text/plain text/css application/x-javascript;

include          /etc/nginx/mime.types;
default_type     application/octet-stream;

server {
    listen [::]:80 ipv6only=off;
    server_name acme.example.org;

    access_log /var/log/nginx/acme.access_log combined;
    error_log /var/log/nginx/acme.error_log info;

    root /var/empty;

    return 302 https://$server_name$request_uri;
}

server {
    listen [::]:443 ipv6only=off http2;
    server_name acme.example.org;

    ssl on;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-
↪SHA:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA;
    ssl_prefer_server_ciphers on;
    ssl_certificate /home/jdoe/acme.crt;
    ssl_certificate_key /home/jdoe/acme.key;

    access_log /var/log/nginx/acme.ssl_access_log combined;
    error_log /var/log/nginx/acme.ssl_error_log info;

    root /var/empty;

    location ~ ^/(images|fonts)(.+)/(.+?)(__v[0-9a-f]+)?\.[^.]$ {
        alias /home/jdoe/dev/indico/src/indico/web/static/$1$2/$3.$5;
    }

    location ~ ^/(css|dist|images|fonts)/(.*)$ {
        alias /home/jdoe/dev/indico/src/indico/web/static/$1/$2;
    }

```

(continues on next page)

(continued from previous page)

```

    }

    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $server_name;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

```

This configuration also assumes you’ve already got a secret key and certificate stored in `~/acme.key` and `acme.crt` respectively. In most cases you will probably use a self-signed certificate. There are many guides on-line on [how to generate a self-signed certificate](#), so we will not cover it here.

If you’re using SELinux, you will need to set the following configuration options:

```

sudo setsebool -P httpd_can_network_connect 1
sudo setsebool -P httpd_read_user_content 1

```

Uploading large files will probably fail unless you do:

```

sudo chown -R jdoe:nginx /var/lib/nginx/tmp/

```

The Indico dev server should be run with the `--proxy` option:

```

indico run -h 127.0.0.1 -p 8000 -q --enable-eval --url https://acme.example.org --
↪proxy

```

You can then start nginx and access `https://acme.example.org` directly.

## 1.1.5 Plugins

We provide a meta-package that contains all official plugins. Before installing it, make sure you are logged in as the *indico* user and inside the Indico environment:

```

su - indico
source ~/.venv/bin/activate

```

Now install the package which will automatically install our plugins:

```

pip install indico-plugins

```

**Note:** Having all plugins installed has no disadvantages; only plugins enabled in `indico.conf` are actually loaded and executed. If you do not use the `indico-plugins` package, we won’t be able to display a notification when updates are available and you would have to update all the plugins separately.

You can use the `indico setup list-plugins` command to see which plugins are installed and which name to use in the config file to load them.

To enable plugins, add a `PLUGINS` entry to `/opt/indico/etc/indico.conf`. For example, the following line would enable the “Bank Transfer” and “PayPal” payment plugins:

```
PLUGINS = {'payment_manual', 'payment_paypal'}
```

Some plugins contain additional database tables. Run the plugin database migrations to create them (if you do not have any plugins with custom tables, the command will simply do nothing):

```
indico db --all-plugins upgrade
```

After any change to the config file, you need to reload uWSGI:

```
touch ~/web/indico.wsgi
```

It is also a good idea to restart the Celery worker (as *root*) since some plugins may come with background tasks:

```
systemctl restart indico-celery.service
```

### 1.1.6 Translations

Indico comes with a number of languages by default. In release 2.3, those are: English (default), French, Portuguese, Spanish and Chinese (in the order of integration). Additional languages are being prepared on the Transifex platform.

In order to use (partially) existing translations from Transifex or to contribute translations, you need to register with the [Indico project on the Transifex platform](#).

#### Additional Translations

This is a guide to set up an Indico instance with a new language. It is useful for translators to verify how the translation looks in production or for administrators who just want to lurk at the incubated translation embryos.

Alternatively, you may use this guide to expose a translation we do not officially support, in your production version.

### 1. Setup an Indico dev environment

This should usually be done on your own computer or a virtual machine.

For creating your own Indico instance, we provide two different guides: The first one is for a *production system*, it will prepare Indico to be served to users and used in all the different purposes you may have besides translations. The second is *development* a light-weight, easier to set up, version oriented to testing purposes, that should not be exposed to the public.

For the purpose of translation **development** or **testing** we recommend using the development version.

### 2. Install the transifex client

Follow the instructions on the [transifex site](#).

### 3. Get an API token

Go to your [transifex settings](#) and generate an API token. Afterwards, you should run the command `tx init --skipsetup`. It will request the token you just copied from the previous settings and save it locally so you can start using transifex locally. If you do not know how to run this command, please refer to the [transifex client guide](#).

## 4. Install the translations

Navigate to `~/dev/indico/src` (assuming you used the standard locations from the dev setup guide).

Run `tx pull -f -l <language_code>`. Languages codes can be obtained [here](#).

For example, Chinese (China) is `zh_CN.GB2312`.

## 5. Compile translations and run Indico

Run the commands `indico i18n compile-catalog` and `indico i18n compile-catalog-react` and:

- *launch Indico*, or
- *build and deploy your own version of Indico*, if you wish to deploy the translation in a production version.

The language should now show up as an option in the top right corner.

In case you modified the `.js` resources, you also need to delete the cached files in `~/dev/indico/data/cache/assets_i18n_*.js`.

## FAQ

### Why isn't Indico loading my language?

Some languages in transifex use codes that Indico is not able to recognize. One example is the Chinese's `zh_CN.GB2312`. The easy fix for this is to rename the folder `zh_CN.GB2312` (inside `indico/translations/`) to the extended locale code `zh_Hant_TW`. Unfortunately, there is no list with mappings for all the languages. So if by any reason it doesn't work for you, feel free to [ask us](#).

## Contributing

As a **translator**, you should have a good knowledge of the Indico functions (from the user side at least). Then you can subscribe to the abovementioned [Transifex site for Indico](#) and request membership of one of the translation teams. You should also contact the coordinators; some languages have specific coordinators assigned. They may point you to places, where work is needed and which rules have been agreed for the translations.

The glossary is usually of big help to obtain a uniform translation of all technical terms. Use it!

As a **programmer** or **developer**, you will have to be aware of the needs and difficulties of translation work. A [Wiki page for Internationalisation](#) is available from github (slightly outdated and we should eventually move it to this documentation). It describes the interface between translating and programming and some conventions to be followed. Everyone involved in translating or programming Indico should have read it before starting the work.

Whenever translators spot difficult code (forgotten pluralization, typos), they should do their best to avoid double (or rather: multiple) work to their fellow translators. What is a problem for their translation, usually will be a problem for all translations. Don't hesitate to open an issue or pull request on [GitHub](#). Repair first, then translate (and be aware that after repair, the translation has to be made again for all languages).

---

**Note:** The codebase also contains legacy code, which may not follow all rules.

---

## File Organisation

The relationship between

- transifex resources names (core.js, core.py, core.react.js)
- PO file names (messages-js.po, messages.po, messages-react.po) and
- the actual place, where the strings are found

is not always obvious. Starting with the resource names, the files ending in

- .py refer to translations used with python and jinja templates,
- .js refer to translations used with generic or legacy javascript,
- react.js refer to translations used with the new react-based javascript.

These contain a relationship to PO files, as defined in the following example extracted from `src/.tx/config`.

```
[indico.<transifex resource slug>]
file_filter = indico/translations/<lang>/LC_MESSAGES/<PO file name>.po
source_file = indico/translations/<source file name>.pot
source_lang = en
type = PO
```

---

**Note:** The transifex resource slug is a name-like alias that identifies a particular file.

---

For more information regarding this subject a [thread has started here](#).

### 1.1.7 LaTeX

Indico uses LaTeX (xelatex to be exact) to generate some PDF files such as the *Book of Abstracts* and the PDF versions of contributions. If you do not need these features, you can skip this part of the documentation and avoid installing LaTeX altogether.

Since Indico requires quite a few LaTeX packages which are not always installed by default when using the texlive packages of the various linux distributions, we recommend installing it manually.

First of all, you will need to install some dependencies so that all TeX formats are generated successfully upon TeXLive installation.

```
yum install fontconfig ghostscript      # CentOS / CC7
apt install libfontconfig1 ghostscript  # Debian / Ubuntu
```

You are now ready to install TeXLive. The following commands should work fine to install everything you need. You need to run the installation as root or create `/opt/texlive` as root and grant your user write access to it.

Download the installer and cd to its location (the directory name contains the date when the package was built, so use the wildcard or type the name manually based on the output when unpacking the archive):

```
cd /tmp
wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz
tar xvfz install-tl-unx.tar.gz
cd install-tl-*/
```

Create the setup config file to install all the packages you need:



```
cat > texlive.profile <<'EOF'
selected_scheme scheme-full
TEXDIR /opt/texlive
TEXMFCONFIG ~/.texlive/texmf-config
TEXMFHOME ~/.texmf
TEXMFLOCAL /opt/texlive/texmf-local
TEXMFSYSCONFIG /opt/texlive/texmf-config
TEXMFSYSVAR /opt/texlive/texmf-var
TEXMFVAR ~/.texlive/texmf-var
binary_x86_64-linux 1
instopt_adjustpath 0
instopt_adjustrepo 0
instopt_letter 0
instopt_portable 0
instopt_writel8_restricted 1
tlpdbopt_autobackup 1
tlpdbopt_backupdir tlpkg/backups
tlpdbopt_create_formats 1
tlpdbopt_generate_updmap 0
tlpdbopt_install_docfiles 0
tlpdbopt_install_srcfiles 0
tlpdbopt_post_code 1
tlpdbopt_sys_bin /usr/local/bin
tlpdbopt_sys_info /usr/local/share/info
tlpdbopt_sys_man /usr/local/share/man
EOF
```

Start the installer and wait for it to complete. This may take between a few minutes and a few hours depending on the speed of the (randomly chosen) mirror.

```
./install-tl --profile texlive.profile
```

After installing it, add this line to your `indico.conf` file to use your new TeXLive installation:

```
XELATEX_PATH = '/opt/texlive/bin/x86_64-linux/xelatex'
```

If you are in a production setup, reload uWSGI using `touch /opt/indico/web/indico.wsgi` to reload the config file.

As security-related updates are released frequently, it is also a good idea to periodically update the TeXLive packages by running:

```
/opt/texlive/bin/x86_64-linux/tlmgr update --self --all
```



Indico is very flexible and many things can be configured/customized in its configuration file.

## 2.1 Configuration

Indico is very flexible and many things can be configured/customized in its configuration file.

### 2.1.1 Settings

`indico.conf` is Indico's main configuration file. Its initial version is usually generated when running `indico setup wizard` as described in the Installation Guide, but depending on the setup it should be modified later.

The config file is loaded from the path specified in the `INDICO_CONFIG` environment variable; if no such path is set, the config file (or a symlink to it) is searched in the following places, in order:

- `<indico_package_path>/indico.conf` (development setups only)
- `~/.indico.conf`
- `/etc/indico.conf`

The file is executed as a Python module, so anything that is valid Python 2.7 code can be used in it. When defining temporary variables that are not config options, their name should be prefixed with an underscore; otherwise you will get a warning about unknowing config options being defined.

### Authentication

#### LOCAL\_IDENTITIES

This setting controls whether local Indico accounts are available. If no centralized authentication infrastructure (e.g. LDAP, OAuth, or another kind of SSO) is used, local accounts are the only way of logging in to Indico.

Default: `True`

#### **LOCAL\_GROUPS**

This setting controls whether local Indico groups are available. If no centralized authentication infrastructure that supports groups (e.g. LDAP) is used, local groups are the only way to define groups in Indico, but if you do have central groups it may be useful to disable local ones to have all groups in one central place.

Default: `True`

#### **LOCAL\_REGISTRATION**

This setting controls whether people accessing Indico can create a new account. Admins can always create new local accounts, regardless of this setting.

This setting is only taken into account if `LOCAL_IDENTITIES` are enabled.

Default: `True`

#### **LOCAL\_MODERATION**

This setting controls whether a new registration needs to be approved by an admin before the account is actually created.

This setting is only taken into account if `LOCAL_IDENTITIES` and `LOCAL_REGISTRATION` are enabled.

Default: `False`

#### **FAILED\_LOGIN\_RATE\_LIMIT**

Applies a rate limit to failed login attempts due to an invalid username or password. When specifying multiple rate limits separated with a semicolon, they are checked in that specific order, which can allow for a short burst of attempts (e.g. a legitimate user trying multiple passwords they commonly use) and then slowing down more strongly (in case someone tries to brute-force more than just a few passwords).

Rate limiting is applied by IP address and only failed logins count against the rate limit. It also does not apply to login attempts using external login systems (SSO) as failures there are rarely related to invalid credentials coming from the user (these would be rejected on the SSO side, which should implement its own rate limiting).

The default allows a burst of 15 attempts, and then only 5 attempts every 15 minutes for the next 24 hours. Setting the rate limit to `None` disables it.

Default: `'5 per 15 minutes; 10 per day'`

#### **EXTERNAL\_REGISTRATION\_URL**

The URL to an external page where people can register an account that can then be used to login to Indico (usually via LDAP/SSO).

This setting is only taken into account if `LOCAL_IDENTITIES` are disabled.

Default: `None`

#### **AUTH\_PROVIDERS**

A dict defining `Flask-Multipass` authentication providers used by Indico. The dict specified here is passed to the `MULTIPASS_AUTH_PROVIDERS` setting of `Flask-Multipass`.

Default: `{ }`

#### **IDENTITY\_PROVIDERS**

A dict defining `Flask-Multipass` identity providers used by Indico to look up user information based on the data provided by an authentication provider. The dict specified here is passed to the `MULTIPASS_IDENTITY_PROVIDERS` setting of `Flask-Multipass`.

Default: `{ }`

#### **PROVIDER\_MAP**

If not specified, authentication and identity providers with the same name are linked automatically. The dict specified here is passed to the `MULTIPASS_PROVIDER_MAP` setting of `Flask-Multipass`.

Default: `{ }`

## Cache

### REDIS\_CACHE\_URL

The URL of the redis server to use for caching.

If the Redis server requires authentication, use a URL like this: `redis://unused:password@127.0.0.1:6379/1`

If no authentication is used (usually the case with a local Redis server), you can omit the user/password part: `redis://127.0.0.1:6379/1`

Default: None

### MEMCACHED\_SERVERS

The list of memcached servers (each entry is an `ip:port` string) to use with the memcached cache backend.

Default: []

## Celery

### CELERY\_BROKER

The URL of the Celery broker (usually Redis or AMQP) used for communication between Indico and the Celery background workers.

We recommend using Redis as it is the easiest option, but you can check the [Celery documentation on brokers](#) for more information on the other possible brokers.

Default: None

### CELERY\_RESULT\_BACKEND

The URL of the Celery result backend. If not set, the same backend as the broker is used. Indico currently does not use task results, and we recommend leaving this setting at its default.

Default: None

### CELERY\_CONFIG

A dict containing additional Celery settings.

**Warning:** This is an advanced setting that is rarely needed and we do not recommend using it unless you know exactly what you are doing! Changing Celery settings may break things or result in tasks not being executed without other changes (such as running additional celery workers on different queues).

One use case for this setting is routing certain tasks to a different queue, and then running multiple Celery workers for these queues.

```
CELERY_CONFIG = {
    'task_routes': {
        'indico_livesync.task.scheduled_update': {'queue': 'livesync'},
    }
}
```

Default: {}

### SCHEDULED\_TASK\_OVERRIDE

A dict overriding the task schedule for specific tasks.

By default, all periodic tasks are enabled and use a schedule which we consider useful for most cases. Using this setting, you can override the default schedule.

The dict key is the name of the task and the value can be one of the following:

- `None` or `False` – disables the task completely
- A dictionary, as described in the [Celery documentation on periodic tasks](#). The task should not be specified, as it is set automatically.
- A `timedelta` or `crontab` object which will just override the schedule without changing any other options of the task. Both classes are available in the config file by default.

---

**Note:** Use `indico celery inspect` registered to get a list of task names. Celery must be running for this command to work.

---

Default: `{}`

## Customization

### CUSTOMIZATION\_DIR

The base path to the directory containing customizations for your Indico instance.

It is possible to override specific templates and add CSS and JavaScript for advanced customizations. When using this, be advised that depending on the modifications you perform things may break after an Indico update. Make sure to test all your modifications whenever you update Indico!

To include custom CSS and JavaScript, simply put `*.css` and `*.js` files into `<CUSTOMIZATION_DIR>/css / <CUSTOMIZATION_DIR>/js`. If there are multiple files, they will be included in alphabetical order, so prefixing them with a number (e.g. `00-base.css`, `10-events.css`) is a good idea.

Static files may be added in `<CUSTOMIZATION_DIR>/files`. They can be referenced in templates through the `assets.custom` endpoint. In CSS/JS, the URL for them needs to be built manually (`/static/custom/files/...`).

For template customizations, see the description of [CUSTOMIZATION\\_DEBUG](#) as this setting is highly recommended to figure out where exactly to put customized templates.

Here is an example for a template customization that includes a custom asset and uses inheritance to avoid having to replace the whole template:

```
{% extends '~footer.html' %}

{% block footer_logo %}
    {%- set filename = 'cern_small_light.png' if dark|default(false) else 'cern_
↪small.png' -%}
    <a href="https://home.cern/" class="footer-logo">
        
    </a>
{% endblock %}
```

Default: `None`

### CUSTOMIZATION\_DEBUG

Whether to log details for all customizable templates the first time they are accessed. The log message contains the path where you need to store the template; this path is relative to `<CUSTOMIZATION_DIR>/templates/`.

The log message also contains the full path of the original template in case you decide to copy it. However, instead of copying templates it is better to use Jinja inheritance where possible. To make this easier the log entry contains a “reference” path that can be used to reference the original template from the customized one.

Default: `False`

#### **HELP\_URL**

The URL used for the “Help” link in the footer.

Default: `'https://learn.getindico.io'`

#### **LOGO\_URL**

The URL to a custom logo. If unset, the default Indico logo is used.

Default: `None`

#### **CUSTOM\_COUNTRIES**

A dict with country name overrides. This can be useful if the official ISO name of a country does not match what your Indico instance’s target audience expects for a country, e.g. due to political situations.

```
CUSTOM_COUNTRIES = {'KP': 'North Korea'}
```

Default: `{}`

#### **CUSTOM\_LANGUAGES**

A dict with language/territory name overrides. This can be useful if the official territory name that goes along with a language does not match what your Indico instance’s target audience expects for a country, e.g. due to political situations.

For example, to replace “Chinese (Simplified)” with “Chinese (China)”, you would use:

```
CUSTOM_LANGUAGES = {'zh_Hans_CN': ('Chinese', 'Simplified')}
```

Note that the language and territory name should be written in that particular language to be consistent with the defaults. So in the example above, you would write “Chinese” and “Simplified” in Simplified Chinese.

Setting the territory (second element in the tuple) to `None` will hide it and only show the language name itself. Setting the dict value to `None` will effectively hide the language altogether.

Default: `{}`

## **Database**

#### **SQLALCHEMY\_DATABASE\_URI**

The URI used to connect to the PostgreSQL database. For a local database, you can usually omit everything besides the database name: `postgresql:///indico`

If the database requires authentication and/or runs on a separate host, this form should be used: `postgresql://user:password@hostname/dbname`

#### **SQLALCHEMY\_POOL\_SIZE**

This setting configures SQLAlchemy’s connection pool. For details, check the [Flask-SQLAlchemy documentation](#).

Default: `5`

#### **SQLALCHEMY\_POOL\_RECYCLE**

This setting configures SQLAlchemy’s connection pool. For details, check the [Flask-SQLAlchemy documentation](#).

Default: `120`

#### **SQLALCHEMY\_POOL\_TIMEOUT**

This setting configures SQLAlchemy’s connection pool. For details, check the [Flask-SQLAlchemy documentation](#).

Default: 10

## Development

**Warning:** Do not turn on development settings in production. While we are not aware of serious security issues caused by these settings, they may slow down Indico or remove redundancies and thus make Indico not as stable as one would expect it to be in a production environment.

### DEBUG

Enables debugging mode. If enabled, assets are not minified, error messages are more verbose and various other features are configured in a developer-friendly way.

**Do not enable debug mode in production.**

Default: `False`

### DB\_LOG

Enables real-time database query logging. When enabled, all database queries are sent to a socket where they can be read by the `db_log.py` script. To use the database logger, run `bin/utils/db_log.py` (only available when running Indico from a Git clone) in a separate terminal and all requests and verbose queries will be displayed there.

Default: `False`

### PROFILE

Enables the Python profiler. The profiler output is stored in `<TEMP_DIR>/*.prof`.

Default: `False`

### SMTP\_USE\_CELERY

If disabled, emails will be sent immediately instead of being handed to a Celery background worker. This is often more convenient during development as you do not need to run a Celery worker while still receiving emails sent from Indico. Disabling it may result in emails not being sent if the mail server is unavailable or some other failure happens during email sending. Because of this, the setting should never be disabled in a production environment.

Default: `True`

### COMMUNITY\_HUB\_URL

The URL of the community hub. This should only be changed when using a local instance of Mereswine to debug the interface between Indico and Mereswine.

Default: `'https://hub.getindico.io'`

### SYSTEM\_NOTICES\_URL

The URL of a YAML file with system notices. This should only be changed during development (to test custom notices) or set to `None` to opt-out from ever fetching or displaying system notices.

Default: `'https://getindico.io/notices.yml'`

### DISABLE\_CELERY\_CHECK

Disables the warning about Celery not running or being outdated. When set to `None`, the warning is disabled when `DEBUG` is enabled; otherwise this setting enables/disables the warning regardless of debug mode.

Default: `None`



## Directories

### **CACHE\_DIR**

The directory in which various data is cached temporarily. Must be accessible by the web server.

Default: `'/opt/indico/cache'`

### **LOG\_DIR**

The directory in which log files are stored. Can be overridden by using absolute paths in `logging.yaml`.

Default: `'/opt/indico/log'`

### **TEMP\_DIR**

The directory in which various temporary files are stored. Must be accessible by the web server.

Default: `'/opt/indico/cache'`

## Emails

### **SMTP\_SERVER**

The hostname and port of the SMTP server used for sending emails.

Default: `('localhost', 25)`

### **SMTP\_LOGIN**

The username to send if the SMTP server requires authentication.

Default: `None`

### **SMTP\_PASSWORD**

The password to send if the SMTP server requires authentication.

Default: `None`

### **SMTP\_USE\_TLS**

If enabled, STARTTLS will be used to use an encrypted SMTP connection.

Default: `False`

### **SMTP\_TIMEOUT**

The timeout in seconds after which a connection attempt to the SMTP server is aborted.

Default: `30`

### **NO\_REPLY\_EMAIL**

The email address used when sending emails to users to which they should not reply.

Default: `None`

### **PUBLIC\_SUPPORT\_EMAIL**

The email address that is shown to users on the “Contact” page.

Default: `None`

### **SUPPORT\_EMAIL**

The email address of the technical manager of the Indico instance. Emails about unhandled errors/exceptions are sent to this address.

Default: `None`

## Experimental Features

### **EXPERIMENTAL\_EDITING\_SERVICE**

If enabled, event managers can connect the Editing module of their events to an external microservice extending the normal Editing workflow. As long as this is considered experimental, there are no guarantees on backwards compatibility even in minor Indico version bumps. Please check the [reference implementation](#) for details/changes.

Default: `False`

## LaTeX

### **XELATEX\_PATH**

The full path to the `xelatex` program of [TeXLive](#).

If it is installed in a directory in your `$PATH`, specifying its name without a path is sufficient.

If the path is not configured, any functionality that requires LaTeX on the server (such as generating the Book of Abstracts or exporting contributions to PDF) will be disabled.

Default: `None`

### **STRICT\_LATEX**

Enables strict mode for LaTeX rendering, in which case a non-zero status code is considered failure.

LaTeX is rather generous when it comes to using a non-zero exit code. For example, having an oversized image in an abstract is enough to cause one. It is generally not a good idea to enable strict mode as this will result in PDF generation to fail instead of creating a PDF that looks slightly uglier (e.g. a truncated image) than one that would succeed without a non-zero status code.

Default: `False`

## Logging

### **LOGGING\_CONFIG\_FILE**

The path to the logging config file. Unless an absolute path is specified, the path is relative to the location of the Indico config file after resolving symlinks.

Default: `'logging.yaml'`

### **SENTRY\_DSN**

If you use [Sentry](#) for logging warnings/errors, you can specify the connection string here.

Default: `None`

### **SENTRY\_LOGGING\_LEVEL**

The minimum level a log record needs to have to be sent to Sentry. If you do not care about warnings, set this to `'ERROR'`.

Default: `'WARNING'`

## Security

### **SECRET\_KEY**

The secret key used to sign tokens in URLs. It must be kept secret under all circumstances.

When using Indico on a cluster of more than one worker, all machines need to have the same secret key.

The initial key is generated by the setup wizard, but if you have to regenerate it, the best way of doing so is running this snippet on a shell: `python -c 'import os; print repr(os.urandom(32))'`

Default: None

#### **SESSION\_LIFETIME**

The duration of inactivity after which a session and its session cookie expires. If set to 0, the session cookie will be cleared when the browser is closed.

Default: 86400 \* 31

## Storage

#### **STORAGE\_BACKENDS**

The list of backends that can be used to store/retrieve files.

Indico needs to store various files such as event attachments somewhere. By default only a filesystem based storage backend is available, but plugins could add additional backends. You can define multiple backends, but once a backend has been used, you **MUST NOT** remove it or all files stored in that backend will become unavailable.

To define a filesystem-based backend, use the string `fs:/base/path`. If you stopped using a backend, you can switch it to read-only mode by using `fs-readonly`: instead of `fs`:

Other backends may accept different options - see the documentation of these backends for details.

Default: `{ 'default': 'fs:/opt/indico/archive' }`

#### **ATTACHMENT\_STORAGE**

The name of the storage backend used to store all kinds of attachments. Anything in this backend is write-once, i.e. once stored, files in it are never modified or deleted.

Changing this only affects new uploads; existing files are taken from the backend that was active when they were uploaded – which is also why you must not remove a backend from [STORAGE\\_BACKENDS](#) once it has been used.

Default: `'default'`

#### **STATIC\_SITE\_STORAGE**

The name of the storage backend used to store “offline copies” of events. Files are written to this backend when generating an offline copy and deleted after a certain amount of time.

If not set, the [ATTACHMENT\\_STORAGE](#) backend is used.

Default: None

## System

#### **BASE\_URL**

This is the URL through which Indico is accessed by users. For production systems this should be an `https://` URL and your web server should redirect all plain HTTP requests to HTTPS.

Default: None

#### **USE\_PROXY**

This setting controls whether Indico runs behind a proxy or load balancer and should honor headers such as `X-Forwarded-For` to get the real IP address of the users accessing it.

The headers taken into account are:

- `X-Forwarded-For` – the IP address of the user

- `X-Forwarded-Proto` – the protocol used by the user
- `X-Forwarded-Host` – the hostname as specified in `BASE_URL` (can be omitted if the `Host` header is correct)

**Warning:** This setting **MUST NOT** be enabled if the server is accessible directly by untrusted clients without going through the proxy or users will be able to spoof their IP address by sending a custom `X-Forwarded-For` header. You need to configure your firewall so only requests coming from your proxy or load balancer are allowed.

Default: `False`

#### **`ROUTE_OLD_URLS`**

If you migrated from an older Indico version (v1.x), enable this option to redirect from the legacy URLs so external links keep working.

Default: `False`

#### **`STATIC_FILE_METHOD`**

This setting controls how static files (like attachments) are sent to clients.

Web servers are very good at doing this; much better and more efficient than Indico or the WSGI container, so this should be offloaded to your web server using this setting.

When using Apache with `mod_xsendfile` or `lighttpd`, set this to `'xsendfile'` and of course enable `xsendfile` in your Apache config.

When using `nginx`, set this to `('xaccelredirect', {'/opt/indico': '/.xsf/indico'})` and add an internal location handler to your `nginx` config to serve `/opt/indico` via `/.xsf/indico`:

```
location /.xsf/indico/ {
    internal;
    alias /opt/indico/;
}
```

The *production installation instructions* already configure this properly, so if you installed Indico using our guide, you only need to change this setting if you add e.g. a new storage backend in `STORAGE_BACKENDS` that stores the files outside `/opt/indico`.

Default: `None`

#### **`MAX_UPLOAD_FILE_SIZE`**

The maximum size of an uploaded file (in MB). A value of 0 disables the limit.

This limit is only enforced on the client side. For a hard limit that is enforced on the server, see `MAX_UPLOAD_FILES_TOTAL_SIZE`

Default: 0

#### **`MAX_UPLOAD_FILES_TOTAL_SIZE`**

The maximum size (in MB) of all files uploaded in a single request (or to be more exact, any data contained in the body of a single request).

A value of 0 disables the limit, but most web servers also have limits which need to be configured as well (`client_max_body_size` in `nginx`) to allow very large uploads.

Default: 0

#### **`DEFAULT_LOCALE`**

The locale that is used by default for i18n. Valid values are `en_GB`, `fr_FR`, and `es_ES`.

Default: 'en\_GB'

#### DEFAULT\_TIMEZONE

The timezone that is used by default. Any timezone identifier such as Europe/Zurich or US/Central can be used.

Default: 'UTC'

#### ENABLE\_ROOMBOOKING

Whether to enable the room booking system.

Default: False

#### PLUGINS

The list of *Indico plugins* to enable.

A list of all installed plugins can be displayed by the `indico setup list-plugins` command; see the guide linked above for details on how to enable plugins.

Default: `set()`

#### CATEGORY\_CLEANUP

This setting specifies categories where events are automatically deleted a certain amount of days after they have been created.

For each entry, the key is the category id and the value the days after which an event is deleted.

**Warning:** This feature is mostly intended for “Sandbox” categories where users test Indico features. Since it is common for such categories to be used for real events nonetheless, we recommend enabling the “Event Header” in the category settings and clearly mention that the event will be deleted after a while.

Default: `{}`

#### WORKER\_NAME

The name of the machine running Indico. The default value is usually fine unless your servers have ugly (e.g. auto-generated) hostnames and you prefer nicer names to show up in error emails.

Default: `socket.getfqdn()`

## 2.1.2 Authentication

Indico uses [Flask-Multipass](#) to handle authentication, searching for users in an external database, and externally managed groups. This means any Flask-Multipass authentication/identity provider can be used in Indico without any modifications to Indico itself.

For a description of the basic settings regarding local accounts (managed within Indico itself), see the [general indico config documentation](#). This guide focuses solely on advanced authentication methods and how to configure them in Indico.

### Configuration

#### Authentication providers

Authentication providers handle the login process, i.e. asking for user credentials or redirecting to an external site in case of SSO.

The `AUTH_PROVIDERS` setting is Indico's equivalent to the `MULTIPASS_AUTH_PROVIDERS` setting of [Flask-Multipass](#).

It must be set to a dict mapping a unique (internal) name of the auth provider (e.g. `mycompany-ldap`) to a dict of whatever data is needed for the given provider.

The following keys are available in the provider data:

- `type` – **Required.** The type of the provider. Valid values are e.g. `ldap`, `authlib`, `shibboleth`, and whatever custom providers you have installed.
- `title` – The title of the provider (shown on the login page). If omitted, the provider name is used.
- `default` – Must be set to `True` for exactly one form-based provider in case more than one such provider is used. The login form of the default provider is displayed when opening the login page so it should be the provider that most people use.
- Any provider-specific settings.

### Identity providers

Identity providers get data about a user who logged in (based on the information passed on by the authentication provider) and also handle searching of external users and groups.

The `IDENTITY_PROVIDERS` setting is Indico's equivalent to the `MULTIPASS_IDENTITY_PROVIDERS` setting of [Flask-Multipass](#).

It must be set to a dict mapping a unique (internal) name of the identity provider (e.g. `mycompany-ldap`) to a dict of whatever data is needed for the given provider. Note that once an identity provider has been used, its name must not be changed.

The following keys are available in the provider data:

- `type` – **Required.** The type of the provider. Valid values are e.g. `ldap`, `authlib`, `shibboleth`, and whatever custom providers you have installed.
- `title` – The title of the provider (shown in the account list of the user profile). If omitted, the provider name is used.
- `trusted_email` – Set this to `True` if all email addresses received from the provider are trustworthy, i.e. if it is guaranteed that an email address actually belongs to the user (either because it's coming from a trusted employee database or the provider is known to send verification emails). If an email is trusted, Indico will use it immediately to start the signup process or associate an existing account with a matching email address. Otherwise a verification email is sent to prove that the user has access to the email address, which is less user-friendly but extremely important to prevent malicious takeovers of Indico accounts.
- `moderated` – Set this to `True` if you want to require manual approval of the registration by an Indico admin. This results in the same workflow as [LOCAL\\_MODERATION](#) in case of local accounts.
- `synced_fields` – This may be set in no more than once identity provider and enables user data synchronization. Its value should be a set of user attributes that can be synchronized during login. The following attributes can be synchronized: `email`, `first_name`, `last_name`, `affiliation`, `phone`, `address`. Due to the unique nature of email addresses, synchronizing them may fail; in that case a warning is displayed and the old email address remains - an Indico admin could merge the users if they are indeed the same person, but this needs to be done manually since merging users is a potentially destructive operation that cannot be undone. It is also strongly recommended to **ONLY** sync emails if the provider has validated emails (ie `trusted_email` set to `True`); otherwise users would get unvalidated (possibly even invalid) emails set on their account during sync.

- `mapping` – A dictionary that maps between keys given by the identity provider and keys expected by Indico for user information. The key of each entry is the Indico-side attribute name; the value is the key under which the data is exposed by the provider. Indico can take user information from the following keys: `first_name`, `last_name`, `email`, `affiliation`, `phone`, `address`. For example, this mapping would use the `givenName` provided by the identity provider to populate the user's `first_name` in Indico:

```
'mapping': {'first_name': 'givenName'}
```

- `identity_info_keys` – By default, all six attributes listed above will be used if the provider has them (either directly or in some other field specified in the `mapping`). If you want to restrict the data from a provider (e.g. because the value it provides is known to be useless/incorrect), you can set this to a set containing only the attributes you want to use. Note that external user search requires email addresses, so if you exclude email addresses here, users from this provider will never appear in search results.
- Any provider-specific settings.

## Links between providers

By default, authentication and identity providers with the same name are linked together. If this is not what you want, you can use the `PROVIDER_MAP` setting to manually link providers. This is useful for advanced cases where you have e.g. both a login form to enter LDAP credentials and a SSO provider, but want to have a single LDAP identity provider that can use the username from either SSO or the LDAP login. In this case you would link both authentication providers to the same identity provider.

## Specific providers

### LDAP

The `ldap` authentication/identity providers are available by default, but to use them you need to install the `python-ldap` library using `pip install python-ldap`.

**Note:** `python-ldap` has some extra system dependencies (`openldap` and `libsasl`). How to install them (`apt`, `yum`, etc.) depends on your linux distribution. The package names are usually `libsasl2-dev` or `libsasl-dev` and `openldap-dev` (or `-devel` on some distros). If one of these libraries is missing, `pip` will fail when installing `python-ldap`. Simply re-run the command after installing the missing library.

Once everything is installed, you can add the LDAP-related settings to your `indico.conf`. Below is an example based on the LDAP config we use at CERN with Active Directory; you can copy this as a starting point for your own config and then adapt it to your own environment:

```
_ldap_config = {
    'uri': 'ldaps://...',
    'bind_dn': 'cn=***,OU=Users,OU=Organic Units,DC=cern,DC=ch',
    'bind_password': '***',
    'timeout': 30,
    'verify_cert': True,
    'page_size': 1500,

    'uid': 'cn',
    'user_base': 'DC=cern,DC=ch',
    'user_filter': '(objectCategory=user)',
```

(continues on next page)

(continued from previous page)

```

    'gid': 'cn',
    'group_base': 'OU=Workgroups, DC=cern, DC=ch',
    'group_filter': '(objectCategory=group)',
    'member_of_attr': 'memberOf',
    'ad_group_style': True
}

AUTH_PROVIDERS = {
    'ldap': {
        'type': 'ldap',
        'title': 'LDAP',
        'ldap': _ldap_config,
        'default': True
    }
}

IDENTITY_PROVIDERS = {
    'ldap': {
        'type': 'ldap',
        'title': 'LDAP',
        'ldap': _ldap_config,
        'mapping': {
            'first_name': 'givenName',
            'last_name': 'sn',
            'email': 'mail',
            'affiliation': 'company',
            'phone': 'telephoneNumber'
        },
        'trusted_email': True,
        'synced_fields': {'first_name', 'last_name', 'affiliation', 'phone', 'address'
    }
}

```

The LDAP-specific config uses the following keys:

- **uri** – **Required.** The URI referring to the LDAP server including the protocol and the port. Use `ldaps://` for LDAP over SSL/TLS and `ldap://` with the `starttls` option for a plain LDAP connection with TLS negotiation. The port can be omitted if the LDAP server listens on the default port (636 for LDAP over SSL and 389 for a plain LDAP connection with TLS negotiation).
- **bind\_dn** – **Required.** The distinguished name to bind to the LDAP directory.
- **bind\_password** – **Required.** The password to use together with the `bind_dn` to login to the LDAP server.
- **timeout** – The delay in seconds to wait for a reply from the LDAP server (set to `-1` to disable). Default: 30
- **verify\_cert** – Whether to verify the TLS certificate of the LDAP server. Default: `True`
- **starttls** – Whether to use STARTTLS to switch to an encrypted connection. Ignored with an `ldaps://` URI. Default: `False`
- **page\_size** – The limit of entries to retrieve at once for a search. 0 means no size limit. It is recommended to have at most the size limit imposed by the server. Default: 1000
- **uid** – The attribute whose value is used as an identifier for the user (typically the username). This attribute must be a single-valued attribute whose value is unique for each user. If the attribute is multi-valued, only the first one retrieved will be returned. Default: `'uid'`



- `user_base` – **Required.** The base node for all the nodes which might contain a user.
- `user_filter` – A valid LDAP filter which will select exclusively all users in the subtree from the `user_base`. The combination of the `user_base` and the `user_filter` must match exclusively all the users. Default: `'(objectClass=person)'`
- `gid` – The attribute whose value is used as an identifier for the group (typically the group's name). This attribute must be a single-valued attribute whose value is unique for each group. If the attribute is multi-valued, only the first one retrieved will be returned. Default: `'cn'`
- `group_base` – **Required.** The base node for all the nodes which might contain a group.
- `group_filter` – A valid LDAP filter which will select exclusively all groups in the subtree from the `group_base`. The combination of the `group_base` and the `group_filter` must match exclusively all the groups. Default: `'(objectClass=groupOfNames)'`
- `member_of_attr` – The multi-valued attribute of a user containing the list of groups the user is a member of. Default: `'memberOf'`

---

**Note:** In case of SLAPD/OpenLDAP, the *member of* attribute must be enabled. While it is not enabled by default, the majority of servers will have it enabled. A simple `ldapsearch` for a user member of any group should show if that is the case. If not, you can check [this article](#) on information how to enable it on your LDAP server. Note that unless you manage the LDAP server, you need to ask the administrator of that server to do that.

---

- `ad_group_style` – Whether the server uses Active-Directory-style groups or not. This is only used when checking if a user is a member of a group. If enabled, the code will take advantage of the `tokenGroups` attribute of a user to check for nested group membership. Otherwise, it will only look through the values of the `member_of_attr`, which should also work for Active Directory, but only for direct membership. Default: `False`

## SAML

The `saml` authentication/identity providers are available by default, but to use them you need to install the `python3-saml` library using `pip install python3-saml`.

---

**Note:** `python3-saml` has some extra system dependencies (`xmlsec`). How to install them (`apt`, `yum`, etc.) depends on your linux distribution. The package name is usually `libxmlsec1-dev` (or `xmlsec1-devel` on RPM-based distros). If this library is missing, `pip` will fail when installing `python3-saml`. Simply re-run the command after installing the missing library.

---

Once everything is installed, you can add the SAML-related settings to your `indico.conf`. Below is an example you can copy to have a good starting point for your own config and then adapt it to your own environment:

```
_saml_config = {
    'sp': {
        'entityId': 'indico-saml',
        # Depending on your security config below you may need to generate
        # a certificate and private key.
        # You can use https://www.samltool.com/self_signed_certs.php or
        # use openssl for it (which is more secure as it ensures the
        # key never leaves your machine)
        'x509cert': '',
        'privateKey': '',
    },
},
```

(continues on next page)

(continued from previous page)

```

'idp': {
    # This metadata is provided by your SAML IdP. You can omit (or
    # leave empty) the whole 'idp' section in case you need SP
    # metadata to register your app and get the IdP metadata from
    # https://indico.example.com/multipass/saml/{auth-provider-name}/metadata
    # and then fill in the IdP metadata afterwards.
    'entityId': 'https://my-idp.example.com',
    'singleSignOnService': {
        'url': 'https://my-idp.example.com/saml',
        'binding': 'urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect'
    },
    'singleLogoutService': {
        'url': 'https://my-idp.example.com/saml',
        'binding': 'urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect'
    },
    'x509cert': ''
},
# These advanced settings allow you to tune the SAML security options.
# Please see the documentation on https://github.com/onelogin/python3-saml
# for details on how they behave. Note that by requiring signatures,
# you usually need to set a cert and key on your SP config.
'security': {
    'nameIdEncrypted': False,
    'authnRequestsSigned': True,
    'logoutRequestSigned': True,
    'logoutResponseSigned': True,
    'signMetadata': True,
    'wantMessagesSigned': True,
    'wantAssertionsSigned': True,
    'wantNameId': True,
    'wantNameIdEncrypted': False,
    'wantAssertionsEncrypted': False,
    'allowSingleLabelDomains': False,
    'signatureAlgorithm': 'http://www.w3.org/2001/04/xmldsig-more#rsa-sha256',
    'digestAlgorithm': 'http://www.w3.org/2001/04/xmlenc#sha256'
}
}

AUTH_PROVIDERS = {
    'saml': {
        'type': 'saml',
        'title': 'SAML SSO',
        'saml_config': _saml_config,
        # If your IdP is using ADFS you may need to uncomment this. For details, see
        # https://github.com/onelogin/python-saml/pull/144
        # 'lowercase_urlencoding': True
    }
}

IDENTITY_PROVIDERS = {
    'saml': {
        'type': 'saml',
        'title': 'SSO',
        'mapping': {
            'first_name': 'Firstname',
            'last_name': 'Lastname',
            'email': 'EmailAddress',
            'affiliation': 'HomeInstitute',

```

(continues on next page)

(continued from previous page)

```

    },
    'trusted_email': True,
    # You can use a different field as the unique identifier.
    # By default the qualified NameID from SAML is used, but in
    # case you want to use something else, any SAML attribute can
    # be used.
    # 'identifier_field': 'Username'
  }
}

```

If you also have an LDAP server, it may be a good idea to use the `saml` authentication provider and connect it to an `ldap` identity provider. This way the user information is retrieved from LDAP based on a unique identifier of the user that comes from SAML, and you can still use the search and group functionality provided by LDAP.

To use this, use the `AUTH_PROVIDERS` config from above together with the `IDENTITY_PROVIDERS` config from the LDAP section on this page, and set up a `PROVIDER_MAP` that passes the identifier from SAML to LDAP. The example below assumes that the LDAP username is passed in a SAML attribute named `UPN`.

```

PROVIDER_MAP = {
    'saml': {'identity_provider': 'ldap', 'mapping': {'identifier': 'UPN'}},
}

```

## Shibboleth

Changed in version 3.0: SAML is now supported without the need for Apache.

---

**Note:** Note that since Indico 3.0 there is a new `saml` auth/identity provider available which does not require Apache/shibd and is thus the recommended option to use regardless of the web server in use.

---

The `shibboleth` authentication/identity providers are available by default, but due to how the protocol works you need to use the Apache webserver to use SAML authentication provider.

You can find guides on how to set it up for [CentOS](#) and [Debian](#).

If you also have an LDAP server, it may be a good idea to use the `shibboleth` authentication provider and connect it to an `ldap` identity provider. This way the user information is retrieved from LDAP based on a unique identifier of the user that comes from SAML, and you can still use the search and group functionality provided by LDAP.



### 3.1 Building

Before starting Indico compilation, this guide assumes you've previously *setup the development base* up until the *configuring step*.

**Warning:** We do not recommend doing these steps on the same system where you are running your production version, as you run into the risk of mixing the latter with development resources.

**Note:** The `master` branch on Git is usually the next version under (heavy) development. Check if there is a `3.*.x` branch for your version and if yes, use that branch instead of `master`.

The first step is to generate a local distribution archive. Navigate to the Indico source folder (by default, it is `~/dev/indico/src`) and run the following command:

```
./bin/maintenance/build-wheel.py indico --add-version-suffix
```

**Note:** The build script refuses to run on a dirty git working directory, so any changes you decide to include must be committed temporarily. You can use `git checkout --detach` to avoid committing to your local master branch; if you plan to actually use the translation the better option would be of course to create a real Git branch.

**Warning:** Make sure you're also not running any other build tool such as `build-assets.py`, as it may interfere with the creation of a production build when running in `--watch` mode.

Finally, the `dist` folder will contain the wheel distribution, the file you should to copy to your production machine:

```
dist/
indico-2.3.1.dev0+202009231923.a14a24f564-py2-none-any.whl
```

To deploy this distribution, you should follow the [production installation guide](#), but instead of installing Indico from PyPI (`pip install indico`), install your custom-built wheel from the previous step:

```
pip install /tmp/indico-2.3.1.dev0+202009231923.a14a24f564-py2-none-any.whl
```

If you already have Indico installed, then simply installing the version from the wheel and restarting uwsgi and indico-celery is all you need to do.

### 3.1.1 Including a new translation

If you are including a new translation, you should also include the moment-js locale in `indico/web/client/js/jquery/index.js` before building:

```
// moment.js locales
import 'moment/locale/your-locale';

import 'moment/locale/zh-cn';
import 'moment/locale/es';
import 'moment/locale/fr';
import 'moment/locale/en-gb';
```

---

**Note:** Put your custom locale first, since `en-gb` needs to be the last one as a fallback.

---

### 4.1 Search

Indico's version 3.0 introduced a brand new reusable and backend-agnostic search module backed up by the SQL storage by default. This module can however be decomposed into a single provider, supporting any external service through a plugin.

Indico provides multiple options for a search service, such as:

- The default SQL based search.
  - A performant and feature-rich ElasticSearch-based search service, [Citadel](#) which can be integrated with Indico easily using the official [Citadel plugin](#).
  - Any external search service, as long as you implement a plugin interface according to the specification below.
- 

#### 4.1.1 Internal Search

The Internal Search is a default SQL based engine implementation, created to support the most basic queries. While not as fast and less feature rich (no filters or aggregations) compared to specialized search engines, this search engine provides a decent option for smaller Indico instances which may not want to spend additional time on deploying a separate service just for search.

It supports the two types of records from a total of six targets:

- Events
  - Categories
  - Contributions
  - Attachments
  - Notes
-

---

**Note:** The Internal Search only supports text-based search on titles, description and notes content.

---

## 4.1.2 External Search Service

Indico provides several powerful features for aggregation and filtering when combined with an external search service supporting them, such as [Citadel](#).

### Aggregations

Aggregations, as seen in [Elastic Search](#), provide a way to combine information in groups according to a certain metric, such as a field value, sum or average.

The screenshot displays the Indico interface. On the left, a 'Category' sidebar lists various groups with checkboxes and counts: Home (113078), Conferences, Workshops and Events (33676), Projects (29379), Conferences (23722), Departments (22411), Groups (14001), Committees (11350), Experiments (10434), Workshops (9629), and Other Committees (7496). The main content area shows a search for 'LHC'. Below the search bar, there are tabs for 'Events (8626)', 'Contributions (32754)', 'Materials (69022)', 'Notes (2676)', and 'Categories'. The 'Events' tab is selected, showing two results. The first result is 'Superconducting magnets for particle accelerators and detectors', with a description mentioning LASA and LHC, and a breadcrumb trail: Home » Schools, Seminars and Courses » Seminars » LHC Seminar. The second result is 'SEMINAR - Steering the field quality of the LHC dipoles', with a description about magnetic field specifications and a breadcrumb trail: Home » Schools, Seminars and Courses » Seminars » Other Seminars.

Indico supports any bucket or metric group, composed of a key, count and filter key:

```
class indico.modules.search.result_schemas.AggregationSchema
    Bases: indico.modules.search.result_schemas._ResultSchemaBase

    Represents an aggregation list.

    buckets = None
        A bucket list representing each group.

    label = None
        The name of the aggregation.

class indico.modules.search.result_schemas.BucketSchema
    Bases: indico.modules.search.result_schemas._ResultSchemaBase

    Represents an individual aggregation bucket element.

    count = None
        The number of elements.

    filter = None
        The key that identifies the element's filter.
```



**key = None**

The aggregation key.

## Filters

Filters act combined upon a certain aggregation on structured data. Consider the following bucket group composed of a single affiliation:

```
{
  "affiliation": {
    "label": "Affiliation",
    "buckets": {
      "key": "CERN",
      "count": 5,
      "filter": "cern"
    }
  }
}
```

The combination of *key* and *filter* from `AggregationSchema` can be used as a way to define a human-readable label to an attribute. A corresponding filter acting upon the same key in the example above would be `affiliation=cern`.

## Placeholders

Placeholders are a special type of filters specifically designed to be part of the user-facing text based search query. Examples of valid placeholders would be: *affiliation:CERN* or *person:"John Doe"*.

Indico expects to receive a list of valid placeholders through `get_placeholders()` where each one will be merely hinted to the user while doing a text based search.

**class** `indico.modules.search.base.IndicoSearchProvider`

Bases: `object`

**get\_placeholders()**

Retrieve the list of search shortcuts that will be shown to users when typing a search query.

**Returns** a list of `SearchOption` instances

### 4.1.3 API Reference

The `IndicoSearchProvider` interface allows an external service to integrate with Indico's search module.

```
class indico.modules.search.base.IndicoSearchProvider
```

Bases: `object`

```
RESULTS_PER_PAGE = 10
```

The number of results to show per page.

```
active = True
```

```
get_placeholders ()
```

Retrieve the list of search shortcuts that will be shown to users when typing a search query.

**Returns** a list of `SearchOption` instances

```
get_sort_options ()
```

Retrieve the list of search sortable options.

**Returns** a list of `SearchOption` instances

```
search (query, user=None, page=None, object_types=(), *, admin_override_enabled=False,
        **params)
```

Search using a custom service across multiple targets.

**Parameters**

- **query** – Keyword based query string
- **user** – The user performing the search (for access checks)
- **page** – The result page to show
- **object\_types** – A filter for a specific `SearchTarget`
- **admin\_override\_enabled** – Whether to ignore access restrictions
- **params** – Any additional search params such as filters

**Returns** a dict with the `ResultSchema` structure

```
class indico.modules.search.base.SearchOption (key: str, label: str)
```

Bases: `object`

```
class indico.modules.search.base.SearchOptions (placeholders: list, sort_options: list)
```

Bases: `object`

```
dump ()
```

```
class indico.modules.search.base.SearchTarget
```

Bases: `int`, `indico.util.enum.IndicoEnum`

An enumeration.

```
attachment = 6
```

```
category = 1
```

```
contribution = 3
```

```
event = 2
```

```
event_note = 5
```

```
subcontribution = 4
```

`indico.modules.search.base.get_search_provider(only_active=True)`

Get the search provider to use for a search.

**Parameters** `only_active` – Whether to check that the provider is active; in case it isn't, the default InternalSearch provider will be used.

## Models

**class** `indico.modules.search.result_schemas.ResultSchemaBase`  
 Bases: `indico.modules.search.result_schemas._ResultSchemaBase`

**category\_path** = None  
 The parent category chain

**class** `indico.modules.search.result_schemas.EventResultSchema`  
 Bases: `indico.modules.search.result_schemas.ResultSchemaBase`

**description** = None  
 The event description

**end\_dt** = None  
 The event end date time

**event\_id** = None  
 The event id

**event\_type** = None  
 The event type

**highlight** = None  
 The event content to highlight

**location** = None  
 The event location

**persons** = None  
 The event associated persons

**start\_dt** = None  
 The event start date time

**title** = None  
 The event title

**type** = None  
 The record type

**class** `indico.modules.search.result_schemas.ContributionResultSchema`  
 Bases: `indico.modules.search.result_schemas.ResultSchemaBase`

**contribution\_id** = None  
 The contribution id

**description** = None  
 The contribution description

**duration** = None  
 The contribution duration

**end\_dt** = None  
 The contribution end date time

```
event_id = None
    The contribution event id

highlight = None
    The contribution content to highlight

location = None
    The contribution location

persons = None
    The contribution associated persons

start_dt = None
    The contribution start date time

title = None
    The contribution title

type = None
    The record type

class indico.modules.search.result_schemas.SubContributionResultSchema
    Bases: indico.modules.search.result_schemas.ContributionResultSchema

    subcontribution_id = None
        The sub-contribution id

    type = None
        The record type

class indico.modules.search.result_schemas.AttachmentResultSchema
    Bases: indico.modules.search.result_schemas.ResultSchemaBase

    attachment_id = None
        The attachment id

    attachment_type = None
        The attachment type

    contribution_id = None
        The attachment contribution id

    event_id = None
        The attachment event id

    filename = None
        The attachment filename

    folder_id = None
        The attachment folder id

    modified_dt = None
        The attachment last modified date time

    subcontribution_id = None
        The attachment sub-contribution id

    title = None
        The attachment title

    type = None
        The record type
```

```
user = None
    The attachment author

class indico.modules.search.result_schemas.EventNoteResultSchema
    Bases: indico.modules.search.result_schemas.ResultSchemaBase

    content = None
        The note content

    contribution_id = None
        The note contribution id

    event_id = None
        The note event id

    highlight = None
        The note content to highlight

    modified_dt = None
        The note last modification date time

    note_id = None
        The note id

    subcontribution_id = None
        The note sub-contribution id

    title = None
        The note title

    type = None
        The record type

    user = None
        The note author

class indico.modules.search.result_schemas.PersonSchema
    Bases: indico.modules.search.result_schemas._ResultSchemaBase

    affiliation = None
        The person's affiliation

    name = None
        The person's name

class indico.modules.search.result_schemas.LocationResultSchema
    Bases: indico.core.marshmallow.IndicoSchema

    address = None
        The address

    room_name = None
        The room name

    venue_name = None
        The venue name

class indico.modules.search.result_schemas.HighlightSchema
    Bases: indico.modules.search.result_schemas._ResultSchemaBase

    content = None
        The field's content to highlight
```

**description = None**

The field's description to highlight

Indico can be extended through plugins, standalone packages of code that do not require any modifications to the Indico core itself. A plugin can perform something very simple such as adding a new command to the Indico CLI to more complex functionalities like introducing new payment methods, chat integration, etc.

We suggest that you first have a look at [Getting started](#) and then head over to the more advance topics in the table of contents.

## 5.1 Extending Indico with plugins

Indico can be extended through plugins, standalone packages of code that do not require any modifications to the Indico core itself. A plugin can perform something very simple such as adding a new command to the Indico CLI to more complex functionalities like introducing new payment methods, chat integration, etc.

We suggest that you first have a look at [Getting started](#) and then head over to the more advance topics in the table of contents.

### 5.1.1 Getting started with Indico plugins

---

**Todo:** Write a **REAL**, simple example of a plugin. Include link to Github repo.

---

#### Example plugin

The following is a minimal plugin that makes use of all capabilities of the plugin API. The **display name** of the plugin is defined by the first line of the docstring and the **description** by the rest of it. The plugin may override signal handlers to hook into Indico and additionally run any initialization needed. For example, it will add some command to Indico CLI, extend the shell context and register some assets. Also, *init* is used to inject CSS and JS bundles outside of the plugin scope.

```

class ExamplePlugin(IndicoPlugin):
    """Example Plugin

    An example plugin that demonstrates the capabilities of the new Indico plugin_
    ↪system.
    """

    settings_form = SettingsForm

    def init(self):
        super(ExamplePlugin, self).init()
        self.inject_bundle('main.js')

    def get_blueprints(self):
        return blueprint

    def add_cli_command(self, manager):
        @manager.command
        @with_plugin_context(self)
        def example():
            """Example command from example plugin"""
            print 'example plugin says hi', current_plugin
            if self.settings.get('show_message'):
                print self.settings.get('dummy_message')

    def extend_shell_context(self, add_to_context):
        add_to_context('bar', name='foo', doc='foobar from example plugin', color=
        ↪'magenta!')

```

The plugin can specify its settings via a IndicoForm:

```

class SettingsForm(IndicoForm):
    dummy_message = StringField('Dummy Message')
    show_message = BooleanField('Show Message')

```

The plugin can also specify request handlers and templates. Templates will be loaded from a *templates* folder within your plugin folder. Your plugin can even load templates from other modules by prefixing the name of the template *'other\_plugin:example'* with *render\_template()*.

```

class WPExample(WPDecorated):
    def _get_body(self, params):
        return render_plugin_template('example.html', **params)

class RHExample(RH):
    def _process(self):
        return WPExample(self, foo=u'bar').display()

class RHTest(RH):
    def _process(self):
        return render_plugin_template('test.html')

blueprint = IndicoPluginBlueprint('example', __name__)
blueprint.add_url_rule('/example', 'example', view_func=RHExample)
blueprint.add_url_rule('/example/x', 'example', view_func=RHExample)

```

(continues on next page)



(continued from previous page)

```
blueprint.add_url_rule('/test', 'test', view_func=RHTest)
```

## 5.1.2 Plugin API reference

Indico's plugin system allows you to extend indico with additional modules which can be installed separately and do not require any modifications to the indico core itself.

**class** `indico.core.plugins.IndicoPlugin` (*plugin\_engine, app*)

Bases: `flask_pluginengine.plugin.Plugin`

Base class for an Indico plugin.

All your plugins need to inherit from this class. It extends the *Plugin* class from Flask-PluginEngine with useful indico-specific functionality that makes it easier to write custom plugins.

When creating your plugin, the class-level docstring is used to generate the friendly name and description of a plugin. Its first line becomes the name while everything else goes into the description.

This class provides methods for some of the more common hooks Indico provides. Additional signals are defined in `signals` and can be connected to custom functions using `connect()`.

**acl\_event\_settings = frozenset()**

A set containing the names of event-specific settings which store ACLs

**acl\_settings = frozenset()**

A set containing the names of settings which store ACLs

**category = None**

The group category that the plugin belongs to

**configurable = False**

If the plugin should link to a details/config page in the admin interface

**default\_event\_settings = {}**

A dictionary containing default values for event-specific settings

**default\_settings = {}**

A dictionary containing default values for settings

**default\_user\_settings = {}**

A dictionary containing default values for user-specific settings

**event\_settings**

`classmethod(function) -> method`

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: @classmethod def f(cls, arg1, arg2, ...):
```

```
...
```

It can be called either on the class (e.g. `C.f()`) or on an instance (e.g. `C().f()`). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

**event\_settings\_converters = {}**

A dict containing custom converters for event-specific settings

**get\_blueprints()**

Return blueprints to be registered on the application.

A single blueprint can be returned directly, for multiple blueprint you need to yield them or return an iterable.

**get\_vars\_js()**

Return a dictionary with variables to be added to vars.js file.

**init()**

Called when the plugin is being loaded/initialized.

If you want to run custom initialization code, this is the method to override. Make sure to call the base method or the other overridable methods in this class will not be called anymore.

**inject\_bundle** (*name*, *view\_class=None*, *subclasses=True*, *condition=None*)

Inject an asset bundle into Indico's pages.

#### Parameters

- **name** – Name of the bundle
- **view\_class** – If a WP class is specified, only inject it into pages using that class
- **subclasses** – also inject into subclasses of *view\_class*
- **condition** – a callable to determine whether to inject or not. only called, when the *view\_class* criterion matches

**inject\_vars\_js()**

Return a string that will define variables for the plugin in the vars.js file.

**settings**

classmethod(function) -> method

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: @classmethod def f(cls, arg1, arg2, ...):
```

```
...
```

It can be called either on the class (e.g. *C.f()*) or on an instance (e.g. *C().f()*). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the *staticmethod* builtin.

**settings\_converters = {}**

A dict containing custom converters for settings

**settings\_form = None**

WTForm for the plugin's settings (requires *configurable=True*). All fields must return JSON-serializable types.

**settings\_form\_field\_opts = {}**

A dictionary which can contain the kwargs for a specific field in the *settings\_form*.

**strict\_settings = True**

If *settings*, *event\_settings* and *user\_settings* should use strict mode, i.e. only allow keys in *default\_settings*, *default\_event\_settings* or *default\_user\_settings* (or the related *acl\_settings* sets). This should not be disabled in most cases; if you need to store arbitrary keys, consider storing a dict inside a single top-level setting.

**template\_hook** (*name*, *receiver*, *priority*=50, *markup*=True)

Register a function to be called when a template hook is invoked.

For details see `register_template_hook()`.

**translation\_domain**

Return the domain for this plugin's translation\_path.

**translation\_path**

Return translation files to be used by the plugin.

By default, get <root\_path>/translations, unless it does not exist.

**user\_settings**

classmethod(function) -> method

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: @classmethod def f(cls, arg1, arg2, ...):
    ...
```

It can be called either on the class (e.g. `C.f()`) or on an instance (e.g. `C().f()`). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

**user\_settings\_converters = {}**

A dict containing custom converters for user-specific settings

**class** `indico.core.plugins.IndicoPluginBlueprint` (*name*, *\*args*, *\*\*kwargs*)

Bases: `flask_pluginengine.mixins.PluginBlueprintMixin`, `indico.web.flask.wrappers.IndicoBlueprint`

The Blueprint class all plugins need to use.

It contains the necessary logic to run the blueprint's view functions inside the correct plugin context and to make the static folder work.

**make\_setup\_state** (*app*, *options*, *first\_registration*=False)

Creates an instance of `BlueprintSetupState()` object that is later passed to the register callback functions. Subclasses can override this to return a subclass of the setup state.

**class** `indico.core.plugins.IndicoPluginBlueprintSetupState` (*blueprint*: *Blueprint*,  
*app*: *Flask*, *options*:  
*Any*, *first\_registration*:  
*bool*)

Bases: `flask_pluginengine.mixins.PluginBlueprintSetupStateMixin`, `indico.web.flask.wrappers.IndicoBlueprintSetupState`

**add\_url\_rule** (*rule*, *endpoint*=None, *view\_func*=None, *\*\*options*)

A helper method to register a rule (and optionally a view function) to the application. The endpoint is automatically prefixed with the blueprint's name.

```
class indico.core.plugins.PluginCategory
```

```
    Bases: str, indico.util.enum.IndicoEnum
```

An enumeration.

```
class indico.core.plugins.WPJinjaMixinPlugin
```

```
    Bases: indico.web.views.WPJinjaMixin
```

```
    static render_template_func (template_name_or_list, **context)
```

Renders a template from the plugin's template folder with the given context.

If the template name contains a plugin name (`pluginname:name`), that name is used instead of the current plugin's name.

#### Parameters

- **template\_name\_or\_list** – the name of the template or an iterable containing template names (the first existing template is used)
- **context** – the variables that should be available in the context of the template.

```
indico.core.plugins.get_plugin_template_module (template_name, **context)
```

Like `get_template_module()`, but using plugin templates

```
indico.core.plugins.plugin_url_rule_to_js (endpoint)
```

Like `url_rule_to_js()` but prepending plugin name prefix to the endpoint

```
indico.core.plugins.url_for_plugin (endpoint, *targets, **values)
```

Like `url_for()` but prepending 'plugin\_' to the blueprint name.

## 5.1.3 Hooking into Indico using Signals

### Contents

- *Hooking into Indico using Signals*
  - *indico.core.signals*
    - \* *indico.core.signals.acl*
    - \* *indico.core.signals.agreements*
    - \* *indico.core.signals.attachments*
    - \* *indico.core.signals.category*
    - \* *indico.core.signals.core*
    - \* *indico.core.signals.event*
    - \* *indico.core.signals.event\_management*
    - \* *indico.core.signals.menu*
    - \* *indico.core.signals.plugin*
    - \* *indico.core.signals.rb*
    - \* *indico.core.signals.rh*
    - \* *indico.core.signals.users*

Signals allow you to hook into certain parts of Indico without adding any code to the core (which is something a plugin can and should not do). Each signal has a *sender* which can be any object (depending on the signal) and possibly some keyword arguments. Some signals also make use of their return value or even require one. Check the documentation of each signal on how it's used.

To avoid breakage with newer versions of Indico, it is highly advised to always accept extra `**kwargs` in your signal receiver. For example, a receiver function could look like this:

```
def receiver(sender, something, **kwargs):
    do_stuff_with(something)
```

## indico.core.signals

### indico.core.signals.acl

#### indico.core.signals.acl.can\_access

Called when *ProtectionMixin.can\_access* is used to determine if a user can access something or not.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *user* and *allow\_admin* arguments of *can\_access* are passed as kwargs with the same name.

The *authorized* argument is *None* when this signal is called at the beginning of the access check and *True* or *False* at the end when regular access rights have already been checked. For expensive checks (such as anything involving database queries) it is recommended to skip the check while *authorized* is *None* since the regular access check is likely to be cheaper (due to ACLs being preloaded etc).

If the signal returns *True* or *False*, the access check succeeds or fails immediately. If multiple subscribers to the signal return contradictory results, *False* wins and access is denied.

#### indico.core.signals.acl.can\_manage

Called when *ProtectionMixin.can\_manage* is used to determine if a user can manage something or not.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *user*, *permission*, *allow\_admin*, *check\_parent* and *explicit\_permission* arguments of *can\_manage* are passed as kwargs with the same name.

If the signal returns *True* or *False*, the access check succeeds or fails without any further checks. If multiple subscribers to the signal return contradictory results, *False* wins and access is denied.

#### indico.core.signals.acl.entry\_changed

Called when an ACL entry is changed.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *User*, *GroupProxy* or *EmailPrincipal* is passed as *principal* and *entry* contains the actual ACL entry (a *PrincipalMixin* instance) or *None* in case the entry was deleted. *is\_new* is a boolean indicating whether the given principal was in the ACL before. If *quiet* is *True*, signal handlers should not perform noisy actions such as logging or sending emails related to the change.

If the ACL uses permissions, *old\_data* will contain a dictionary of the previous permissions (see *PrincipalPermissionsMixin.current\_data*).

#### indico.core.signals.acl.get\_management\_permissions

Expected to return *ManagementPermission* subclasses. The *sender* is the type of the object the permissions may be used for. Functions subscribing to this signal **MUST** check the sender by specifying it using the first argument of *connect\_via()* or by comparing it inside the function.

#### indico.core.signals.acl.protection\_changed

Called when the protection mode of an object is changed.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The old protection mode is passed as *old\_mode*, the new mode as *mode*.

### **indico.core.signals.agreements**

`indico.core.signals.agreements.get_definitions`

Expected to return a list of `AgreementDefinition` classes.

### **indico.core.signals.attachments**

`indico.core.signals.attachments.attachment_accessed`

Called when an attachment is accessed. The *sender* is the *Attachment* that was accessed. The user who accessed the attachment is passed in the *user* kwarg. The *from\_preview* kwarg will be set to `True` if the download link on the preview page was used to access the attachment or if the attachment was loaded to be displayed on the preview page (opening the preview itself already sends this signal with *from\_preview=False*).

`indico.core.signals.attachments.attachment_created`

Called when a new attachment is created. The *sender* object is the new *Attachment*. The user who created the attachment is passed in the *user* kwarg.

`indico.core.signals.attachments.attachment_deleted`

Called when an attachment is deleted. The *sender* object is the *Attachment* that was deleted. The user who deleted the attachment is passed in the *user* kwarg.

`indico.core.signals.attachments.attachment_updated`

Called when an attachment is updated. The *sender* is the *Attachment* that was updated. The user who updated the attachment is passed in the *user* kwarg.

`indico.core.signals.attachments.folder_created`

Called when a new attachment folder is created. The *sender* is the new *AttachmentFolder* object. The user who created the folder is passed in the *user* kwarg. This signal is never triggered for the internal default folder.

`indico.core.signals.attachments.folder_deleted`

Called when a folder is deleted. The *sender* is the *AttachmentFolder* that was deleted. The user who deleted the folder is passed in the *user* kwarg.

`indico.core.signals.attachments.folder_updated`

Called when a folder is updated. The *sender* is the *AttachmentFolder* that was updated. The user who updated the folder is passed in the *user* kwarg.

`indico.core.signals.attachments.get_file_previewers`

Expected to return one or more *Previewer* subclasses.

### **indico.core.signals.category**

`indico.core.signals.category.created`

Called when a new category is created. The *sender* is the new category.

`indico.core.signals.category.deleted`

Called when a category is deleted. The *sender* is the category.

`indico.core.signals.category.extra_events`

Called when a category is displayed. The *sender* is the category. *is\_flat* is passed as kwarg with the same name. The additional kwargs passed to this signal depend on the context.

`indico.core.signals.category.moved`

Called when a category is moved into another category. The *sender* is the category and the old parent category is passed in the *old\_parent* kwarg.

`indico.core.signals.category.updated`

Called when a category is modified. The *sender* is the updated category.

## **indico.core.signals.core**

`indico.core.signals.core.add_form_fields`

Lets you add extra fields to a form. The *sender* is the form class and should always be specified when subscribing to this signal.

The signal handler should return one or more 'name', `Field` tuples. Each field will be added to the form as `ext__<name>` and is automatically excluded from the form's *data* property and its *populate\_obj* method.

To actually process the data, you can use e.g. the *form\_validated* signal and then store it in *flask.g* until another signal informs you that the operation the user was performing has been successful.

`indico.core.signals.core.after_commit`

Called after an SQL transaction has been committed. Note that the session is in 'committed' state when this signal is called, so no SQL can be emitted while this signal is being handled.

`indico.core.signals.core.after_process`

Called after an Indico request has been processed. This signal should also be triggered by CLI utilities that result in other signals being triggered.

`indico.core.signals.core.app_created`

Called when the app has been created. The *sender* is the flask app.

`indico.core.signals.core.before_notification_send`

Executed before a notification is sent. The *sender* is a string representing the type of notification. The notification email that will be sent is passed in the *email* kwarg. The additional kwargs passed to this signal depend on the context.

`indico.core.signals.core.check_password_secure`

Check whether a password is secure. The *sender* is a string indicating the context where the password check happens, the plaintext password is sent in the *password* kwarg. To fail the security check for a password, the signal handler should return a string describing why the password is not secure.

`indico.core.signals.core.db_schema_created`

Executed when a new database schema is created. The *sender* is the name of the schema.

`indico.core.signals.core.form_validated`

Triggered when an IndicoForm was validated successfully. The *sender* is the form object.

This signal may return `False` to mark the form as invalid even though WTForms validation was successful. In this case it is highly recommended to mark a field as erroneous or indicate the error in some other way.

`indico.core.signals.core.get_conditions`

Expected to return one or more classes inheriting from *Condition*. The *sender* is a string (or some other object) identifying the context. The additional kwargs passed to this signal depend on the context.

`indico.core.signals.core.get_fields`

Expected to return *BaseField* subclasses. The *sender* is an object (or just a string) identifying for what to get fields. This signal should never be registered without restricting the sender to ensure only the correct field types are returned.

**indico.core.signals.core.get\_placeholders**

Expected to return one or more *Placeholder* objects. The *sender* is a string (or some other object) identifying the context. The additional kwargs passed to this signal depend on the context.

**indico.core.signals.core.get\_search\_providers**

Expected to return exactly one *IndicoSearchProvider* subclass. No more than one handler for this signal may return one as using multiple search providers at the same time is not possible.

**indico.core.signals.core.get\_storage\_backends**

Expected to return one or more *Storage* subclasses.

**indico.core.signals.core.import\_tasks**

Called when Celery needs to import all tasks. Use this signal if you have modules containing task registered using one of the Celery decorators but don't import them anywhere. The signal handler should only import these modules and do nothing else.

**indico.core.signals.event****indico.core.signals.event.abstract\_created**

Called when a new abstract is created. The *sender* is the new abstract.

**indico.core.signals.event.abstract\_deleted**

Called when an abstract is deleted. The *sender* is the abstract.

**indico.core.signals.event.abstract\_state\_changed**

Called when an abstract is withdrawn. The *sender* is the abstract.

**indico.core.signals.event.abstract\_updated**

Called when an abstract is modified. The *sender* is the abstract.

**indico.core.signals.event.after\_registration\_form\_clone**

Executed after a registration form is cloned. The sender is the old *RegistrationForm* object being cloned. The new *RegistrationForm* object is passed in the *new\_form* kwarg.

**indico.core.signals.event.before\_check\_registration\_email**

Called before checking the validity of the registration email. The sender is the *RegistrationForm* object. The signal handler is expected to return *None* if all checks passed or a {'status': ..., 'conflict': ...} dictionary. 'status' is expected to be either 'error', 'warning' or ok.

**indico.core.signals.event.cloned**

Called when an event is cloned. The *sender* is the *Event* object of the old event, the new event is passed in the *new\_event* kwarg.

**indico.core.signals.event.contribution\_created**

Called when a new contribution is created. The *sender* is the new contribution.

**indico.core.signals.event.contribution\_deleted**

Called when a contribution is deleted. The *sender* is the contribution.

**indico.core.signals.event.contribution\_updated**

Called when a contribution is modified. The *sender* is the contribution. A dict containing old, new tuples for all changed values is passed in the *changes* kwarg.

**indico.core.signals.event.created**

Called when a new event is created. The *sender* is the new *Event*. The *cloning* kwarg indicates whether the event is a clone.

**indico.core.signals.event.deleted**

Called when an event is deleted. The *sender* is the event object. The *user* kwarg contains the user performing the deletion if available.



**indico.core.signals.event.filter\_selectable\_badges**

Called when composing lists of badge templates. The *sender* may be either `BadgeSettingsForm`, `RHListEventTemplates` or `RHListCategoryTemplates`. The list of badge templates is passed in the *badge\_templates* kwarg. The signal handler is expected to mutate the list.

**indico.core.signals.event.generate\_ticket\_qr\_code**

Called when generating the QR code for a ticket. The data included in the QR code is passed in the *ticket\_data* kwarg and may be modified.

**indico.core.signals.event.get\_feature\_definitions**

Expected to return *EventFeature* subclasses.

**indico.core.signals.event.get\_log\_renderers**

Expected to return *EventLogRenderer* classes.

**indico.core.signals.event.imported**

Called when data is imported to an event. The *sender* is the *Event* data was imported into, the source event is passed in the *source\_event* kwarg.

**indico.core.signals.event.is\_ticket\_blocked**

Called when resolving whether Indico should let a registrant download their ticket. The *sender* is the registrant's *Registration* object.

If this signal returns `True`, the user will not be able to download their ticket. Any badge containing a ticket-specific placeholder such as the ticket qr code is considered a ticket, and the restriction applies to both users trying to get their own ticket and managers trying to get a ticket for a registrant.

**indico.core.signals.event.is\_ticketing\_handled**

Called when resolving whether Indico should send tickets with e-mails or it will be handled by other module. The *sender* is the *RegistrationForm* object.

If this signal returns `True`, no ticket will be emailed on registration.

**indico.core.signals.event.location\_changed**

Called when the location of an object changed. The *sender* is the type of the object, the object itself is passed as *obj*. The changes are passed in the *changes* kwarg.

**indico.core.signals.event.metadata\_postprocess**

Called right after a dict-like representation of an event is created, so that plugins can add their own fields.

The *sender* is a string parameter specifying the source of the metadata. The *event* kwarg contains the event object. The metadata is passed in the *data* kwarg. The *user* kwarg contains the user for whom the data is generated.

The signal should return a dict that will be used to update the original representation (fields to add or override).

**indico.core.signals.event.moved**

Called when an event is moved to a different category. The *sender* is the event, the old category is in the *old\_parent* kwarg.

**indico.core.signals.event.note\_added**

Called when a note is added. The *sender* is the note.

**indico.core.signals.event.note\_deleted**

Called when a note is deleted. The *sender* is the note.

**indico.core.signals.event.note\_modified**

Called when a note is modified. The *sender* is the note.

**indico.core.signals.event.note\_restored**

Called when a previously-deleted note is restored. The *sender* is the note. This is triggered when a “new” note is created on an object that previously already had a note which got deleted.

`indico.core.signals.event.person_updated`

Called when an `EventPerson` is modified. The *sender* is the `EventPerson`.

`indico.core.signals.event.print_badge_template`

Called when printing a badge template. The registration form is passed in the *regform* kwarg. The list of registration objects are passed in the *registrations* kwarg and it may be modified.

`indico.core.signals.event.registration_checkin_updated`

Called when the checkin state of a registration changes. The *sender* is the `Registration` object.

`indico.core.signals.event.registration_created`

Called when a new registration has been created. The *sender* is the `Registration` object. The *data* kwarg contains the form data used to populate the registration fields. The *management* kwarg is set to `True` if the registration was created from the event management area.

`indico.core.signals.event.registration_deleted`

Called when a registration is removed. The *sender* is the `Registration` object.

`indico.core.signals.event.registration_form_created`

Called when a new registration form is created. The *sender* is the `RegistrationForm` object.

`indico.core.signals.event.registration_form_deleted`

Called when a registration form is removed. The *sender* is the `RegistrationForm` object.

`indico.core.signals.event.registration_form_edited`

Called when a registration form is edited. The *sender* is the `RegistrationForm` object.

`indico.core.signals.event.registration_form_wtform_created`

Called when a the wtform is created for rendering/processing a registration form. The sender is the `RegistrationForm` object. The generated `WTForm` class is passed in the *wtform\_cls* kwarg and it may be modified. The *registration* kwarg contains a `Registration` object when called from registration edit endpoints. The *management* kwarg is set to `True` if the registration form is rendered/processed from the event management area.

`indico.core.signals.event.registration_personal_data_modified`

Called when the registration personal data is modified. The *sender* is the `Registration` object; the change is passed in the *change* kwarg.

`indico.core.signals.event.registration_state_updated`

Called when the state of a registration changes. The *sender* is the `Registration` object; the previous state is passed in the *previous\_state* kwarg.

`indico.core.signals.event.registration_updated`

Called when a registration has been updated. The *sender* is the `Registration` object. The *data* kwarg contains the form data used to populate the registration fields. The *management* kwarg is set to `True` if the registration was updated from the event management area.

`indico.core.signals.event.restored`

Called when a previously-deleted event is restored. The *sender* is the event object. The *user* kwarg contains the user restoring the event if available, and the *reason* kwarg the reason if available.

`indico.core.signals.event.session_block_deleted`

Called when a session block is deleted. The *sender* is the session block. This signal is called before the `db.session.delete()` on the block is executed.

`indico.core.signals.event.session_block_updated`

Called when a session block is updated. The *sender* is the session block.

`indico.core.signals.event.session_deleted`

Called when a session is deleted. The *sender* is the session.

`indico.core.signals.event.session_updated`

Called when a session is updated. The *sender* is the session.

`indico.core.signals.event.sidemenu`

Expected to return `MenuEntryData` objects to be added to the event side menu. A single entry can be returned directly, multiple entries must be yielded.

`indico.core.signals.event.subcontribution_created`

Called when a new subcontribution is created. The *sender* is the new subcontribution.

`indico.core.signals.event.subcontribution_deleted`

Called when a subcontribution is deleted. The *sender* is the subcontribution.

`indico.core.signals.event.subcontribution_updated`

Called when a subcontribution is modified. The *sender* is the subcontribution.

`indico.core.signals.event.times_changed`

Called when the times of a scheduled object (contribution, break or session block) change, either by a change in duration or start time. The *sender* is the type of the object; the timetable entry is passed as *entry* and the object is passed as *obj*. Information about the changes are passed as *changes* which is a dict containing old/new tuples for *start\_dt*, *duration* and *end\_dt*. If an attribute did not change, it is not included in the dict. If the time of the event itself changes, *entry* is `None` and *obj* contains the *Event*.

`indico.core.signals.event.timetable_buttons`

Expected to return a list of tuples ('button\_name', 'js-call-class'). Called when building the timetable view.

`indico.core.signals.event.timetable_entry_created`

Called when a new timetable entry is created. The *sender* is the new entry.

`indico.core.signals.event.timetable_entry_deleted`

Called when a timetable entry is deleted. The *sender* is the entry. This signal is triggered right before the entry deletion is performed.

`indico.core.signals.event.timetable_entry_updated`

Called when a timetable entry is updated. The *sender* is the entry. A dict containing old, new tuples for all changed values is passed in the *changes* kwarg.

`indico.core.signals.event.type_changed`

Called when the type of an event is changed. The *sender* is the event, the old type is passed in the *old\_type* kwarg.

`indico.core.signals.event.updated`

Called when basic data of an event is updated. The *sender* is the event. A dict of changes is passed in the *changes* kwarg, with (old, new) tuples for each change. Note that the *person\_links* change may happen with *old* and *new* being the same lists for technical reasons. If the key is present, it should be assumed that something changed (usually the order or some data on the person link).

## **indico.core.signals.event\_management**

`indico.core.signals.event_management.get_cloners`

Expected to return one or more `EventCloner` subclasses implementing a cloning operation for something within an event.

`indico.core.signals.event_management.image_created`

Called when a new image is created. The *sender* object is the new `ImageFile`. The user who uploaded the image is passed in the *user* kwarg.

`indico.core.signals.event_management.image_deleted`

Called when an image is deleted. The *sender* object is the `ImageFile` that is about to be deleted. The user who uploaded the image is passed in the *user* kwarg.

`indico.core.signals.event_management.management_url`

Expected to return a URL for the event management page of the plugin. This is used when someone who does not have event management access wants to go to the event management area. He is then redirected to one of the URLs returned by plugins, i.e. it is not guaranteed that the user ends up on a specific plugin's management page. The signal should return `None` if the current user (available via `session.user`) cannot access the management area. The *sender* is the event object.

## `indico.core.signals.menu`

`indico.core.signals.menu.items`

Expected to return one or more *SideMenuItem* to be added to the side menu. The *sender* is an id string identifying the target menu.

`indico.core.signals.menu.sections`

Expected to return one or more *SideMenuSection* objects to be added to the side menu. The *sender* is an id string identifying the target menu.

## `indico.core.signals.plugin`

`indico.core.signals.plugin.cli`

Expected to return one or more click commands/groups. If they use *indico.cli.core.cli\_command* / *indico.cli.core.cli\_group* they will be automatically executed within a plugin context and run within a Flask app context by default.

`indico.core.signals.plugin.get_blueprints`

Expected to return one or more IndicoPluginBlueprint-based blueprints which will be registered on the application. The Blueprint must be named either *PLUGINNAME* or *compat\_PLUGINNAME*.

`indico.core.signals.plugin.get_conference_themes`

Expected to return (name, css, title) tuples for conference stylesheets. *name* is the internal name used for the stylesheet which will be stored when the theme is selected in an event. *css* is the location of the CSS file, relative to the plugin's *static* folder. *title* is the title displayed to the user when selecting the theme.

`indico.core.signals.plugin.get_event_request_definitions`

Expected to return one or more RequestDefinition subclasses.

`indico.core.signals.plugin.get_event_themes_files`

Expected to return the path of a themes yaml containing event theme definitions.

`indico.core.signals.plugin.get_template_customization_paths`

Expected to return the absolute path to a directory containing template overrides. This signal is called once during initialization so it should not use any data that may change at runtime. The behavior of a customization path returned by this function is exactly like `<CUSTOMIZATION_DIR>/templates`, but it has lower priority than the one from the global customization dir.

`indico.core.signals.plugin.inject_bundle`

Expected to return a list of bundle names which are loaded after all the rest. The *sender* is the WP class of the page.

`indico.core.signals.plugin.schema_post_dump`

Called when a marshmallow schema is dumped. The *sender* is the schema class and code using this signal should always specify it. The signal is called with the following arguments:

- **many** – bool indicating whether the data was dumped with `many=True` or not
- **data** – the dumped data. this is guaranteed to be a list; in case of `many=False` it is guaranteed to contain exactly one element

- `orig` – the original data before dumping. just like `data` it is always a list

If a plugin wants to modify the data returned when dumping, it may do so by modifying the contents of `data`.

#### `indico.core.signals.plugin.schema_post_load`

Called after a marshmallow schema is loaded. The *sender* is the schema class and code using this signal should always specify it. The signal is called with the following arguments:

- **data** – the data returned by marshmallow; this is usually a dict which may contain more complex data types than those valid in JSON

If a plugin wants to modify the resulting data, it may do so by modifying the contents of `data`.

#### `indico.core.signals.plugin.schema_pre_load`

Called when a marshmallow schema is loaded. The *sender* is the schema class and code using this signal should always specify it. The signal is called with the following arguments:

- **data** – the raw data passed to marshmallow; this is usually a dict of raw json/form data coming from the user, so it can have all types valid in JSON

If a plugin wants to modify the data the schema will eventually load, it may do so by modifying the contents of `data`.

#### `indico.core.signals.plugin.shell_context`

Called after adding stuff to the *indico shell* context. Receives the *add\_to\_context* and *add\_to\_context\_multi* keyword args with functions which allow you to add custom items to the context.

#### `indico.core.signals.plugin.template_hook`

Expected to return a (*is\_markup*, *priority*, *value*) tuple. The returned value will be inserted at the location where this signal is triggered; if multiple receivers are connected to the signal, they will be ordered by priority. If *is\_markup* is True, the value will be wrapped in a *Markup* object which will cause it to be rendered as HTML. The *sender* is the name of the actual hook. The keyword arguments depend on the hook.

### `indico.core.signals.rb`

#### `indico.core.signals.rb.booking_created`

Executed after a booking has been successfully created. The *sender* is the new *Reservation* object.

#### `indico.core.signals.rb.booking_deleted`

Executed after a booking has been deleted. The *sender* is the *Reservation* object.

#### `indico.core.signals.rb.booking_modified`

Executed after a booking has been modified. The *sender* is the *Reservation* object and a dictionary of changed values is passed in the *changes* kwarg.

#### `indico.core.signals.rb.booking_occurrence_state_changed`

Executed after the state of a booking occurrence changed. The *sender* is the *ReservationOccurrence* object.

#### `indico.core.signals.rb.booking_state_changed`

Executed after a booking has been cancelled/rejected/accepted. The *sender* is the *Reservation* object.

### `indico.core.signals.rh`

#### `indico.core.signals.rh.before_process`

Executed right before *\_process* of an *RH* instance is called. The *sender* is the *RH* class, the current instance is passed in *rh*. If a signal handler returns a value, the original *\_process* method will not be executed. If multiple signal handlers return a value, an exception is raised.

**indico.core.signals.rh.check\_access**

Executed right after `_check_access` of an *RH* instance has been called unless the access check raised an exception. The *sender* is the RH class, the current instance is passed in *rh*.

**indico.core.signals.rh.process**

Executed right after `_process` of an *RH* instance has been called. The *sender* is the RH class, the current instance is passed in *rh*. The return value of `_process` is available in *result* and if a signal handler returns a value, it will replace the original return value. If multiple signals handlers return a value, an exception is raised.

**indico.core.signals.rh.process\_args**

Executed right after `_process_args` of an *RH* instance has been called. The *sender* is the RH class, the current instance is passed in *rh*. The return value of `_process_args` (usually `None`) is available in *result*.

**indico.core.signals.users****indico.core.signals.users.email\_added**

Called when a new email address is added to a user. The *sender* is the user object and the email address is passed in the *email* kwarg. The *silent* kwarg indicates whether the email was added during some automated process where no messages should be flashed (e.g. because the sync was in a background task or triggered during a request from another user).

**indico.core.signals.users.logged\_in**

Called when a user logs in. The *sender* is the User who logged in. Depending on whether this was a regular login or an admin impersonating the user, either the *identity* kwarg is set to the *Identity* used by the user to log in or the *admin\_impersonation* kwarg is `True`.

**indico.core.signals.users.merged**

Called when two users are merged. The *sender* is the main user while the merged user (i.e. the one being deleted in the merge) is passed via the *source* kwarg.

**indico.core.signals.users.preferences**

Expected to return a *ExtraUserPreferences* subclass which implements extra preferences for the user preference page. The *sender* is the user for whom the preferences page is being shown which might not be the currently logged-in user!

**indico.core.signals.users.primary\_email\_changed**

Called when the primary address is changed. The *sender* is the user object and the *new* and *old* values are passed as kwargs.

**indico.core.signals.users.registered**

Called once a user registers (either locally or joins through a provider). The *sender* is the new user object. The kwarg *from\_moderation* indicates whether the user went through a moderation process (this also includes users created by an administrator manually) or was created immediately on registration; the identity associated with the registration is passed in the *identity* kwarg.

**indico.core.signals.users.registration\_requested**

Called when a user requests to register a new indico account, i.e. if moderation is enabled. The *sender* is the registration request.

## 5.1.4 Adding models to your plugin

Plugins must describe its database model the in the *models* folder if needed:

```
class Foo(db.Model):
    __tablename__ = 'foo'
    __table_args__ = {'schema': 'plugin_example'}
```

(continues on next page)

(continued from previous page)

```

id = db.Column(
    db.Integer,
    primary_key=True
)
bar = db.Column(
    db.String,
    nullable=False,
    default=''
)
location_id = db.Column(
    db.Integer,
    db.ForeignKey('roombooking.locations.id'),
    nullable=False
)
location = db.relationship(
    'Location',
    backref=db.backref('example_foo', cascade='all, delete-orphan', lazy='dynamic
↪'),
)

def __repr__(self):
    return u'<Foo({}, {}, {})>'.format(self.id, self.bar, self.location)

```

Thanks to **Alembic**, the migration needed to create the tables in the database can also be included in the plugin. The steps to do so are:

1. Create a revision for the changes your plugin will add with `indico db --plugin example migrate -m 'short description'`
2. Fine-tune the revision file generated under *migrations*.
3. Run `indico db --plugin example upgrade` to have Alembic upgrade your DB with the changes.





Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

## 6.1 Indico - HTTP API

Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

### 6.1.1 Accessing the API

#### URL structure

Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

The basic URL looks like:

```
https://my.indico.server/export/WHAT/[LOC/]ID.TYPE?PARAMS
```

or when using legacy API keys:

```
https://my.indico.server/export/WHAT/[LOC/]ID.TYPE?PARAMS&ak=KEY&timestamp=TS&signature=SI
```

where:

- *WHAT* is the element you want to export (one of *categ*, *event*, *room*, *reservation*)
- *LOC* is the location of the element(s) specified by *ID* and only used for certain elements, for example, for the room booking (<https://indico.server/export/room/CERN/120.json?ak=0...>)
- *ID* is the ID of the element you want to export (can be a - separated list). As for example, the 120 in the above URL.
- *TYPE* is the output format (one of *json*, *jsonp*, *xml*, *html*, *ics*, *atom*, *bin*)

- *PARAMS* are various parameters affecting (filtering, sorting, ...) the result list
- *KEY*, *TS*, *SIG* are part of the *API Key Authentication (Deprecated)*

Some examples could be:

- Export data about events in a category: `/export/categ/2.json?from=today&to=today&pretty=yes`
- Export data about a event: `/export/event/137346.json?occ=yes&pretty=yes`
- Export data about rooms: `/export/room/CERN/120.json?pretty=yes`
- Export your reservations: `/export/reservation/CERN.json?detail=reservations&from=today&to=today`

See more details about querying in [Exporters](#).

### API Token Authentication

New in version 3.0.

Indico users may create API tokens with a custom name and scope. They can then be used to authenticate requests to the Indico API using the standard `Authorization: Bearer <token>` HTTP header.

Compared to the legacy API key authentication (see below), they have various advantages:

- no need to generate signatures and deal with expiring links - nowadays with HTTPS being widespread, the risk of leaking a link (but not the secrets used to generate it) is very low
- authentication using a HTTP header avoids including sensitive information in the query string
- each application/script can get its own token, which can have only the scopes assigned that are actually needed
- they behave exactly like OAuth tokens, except that no OAuth application or OAuth flow is required, which makes them perfect for use in custom scripts

These personal API tokens always have the format `indp_<42 random chars>` - tokens generated during a regular OAuth flow have the `indo_` prefix instead.

---

**Note:** Indico administrators have the ability to restrict the creation of API tokens; in that case only admins can create tokens or manage their scopes, but users who have a token can still reset it in order to use the API once authorized by an admin.

---

### Scopes

API tokens can have one of these scopes:

- `full:everything` - Everything (all methods)
- `read:everything` - Everything (only GET)
- `read:legacy_api` - Classic API (read only)
- `write:legacy_api` - Classic API (write only)
- `registrants` - Event registrants
- `read:user` - User information (read only)

The `everything` scopes are special because they can be used with *any* Indico endpoint, i.e. they are not restricted to official APIs. This has the advantage that even Indico actions which do not have a corresponding API can be scripted. Endpoints covered by the `legacy_api` scopes are *not* included; these scopes need to be granted explicitly.

**Warning:** We make absolutely no promises of backwards compatibility on endpoints that are not part of documented APIs. You use them at your own risk.

The `legacy_api` scopes grant access to the API this documentation is about, i.e. `/export/` for retrieving data and some `/api/` paths for modifying data.

The `read:user` scope grants access to basic information about the current user via the `/api/user/` endpoint:

```
{
  "admin": false,
  "email": "guinea.pig@example.com",
  "first_name": "Guinea",
  "id": 1337,
  "last_name": "Pig"
}
```

The `registrants` scope is mainly used by the mobile check-in app and grants access to (currently) undocumented APIs that allow retrieving the list of registrants in an event and updating their check-in state.

## API Key Authentication (Deprecated)

Deprecated since version 3.0: Use [API Token Authentication](#) instead. This authentication method may be removed in a future version.

## General

The HTTP Export API uses an API key and - depending on the config - a cryptographic signature for each request.

To create an API key, go to *My Profile » HTTP API* and click the *Create API key* button. This will create an *API Key* and a *Secret Key* (if signatures are required).

It is recommended to always use the highest security level. That means if only an *API key* is available always include it and if a *secret key* is available, always sign your requests. Since you might want to retrieve only public information (instead of everything visible to your Indico user) you can add the param `onlypublic=yes` to the query string.

It is also possible to re-use the existing Indico session. This only makes sense if your browser accesses the API, e.g. because you are developing on Indico and want to access the API via an AJAX request. Additionally this method of authentication is restricted to GET requests. To use it, add `cookieauth=yes` to the query string and do not specify an API key, timestamp or signature. To prevent data leakage via CSRF the CSRF token of the current session needs to be provided as a GET argument `csrftoken` or a HTTP header `X-CSRF-Token`.

## Request Signing

To sign a request, you need the following:

- The requested path, e.g. `/export/categ/123.json`
- Any additional params, e.g. `limit=10`
- The current UNIX timestamp

- Your *API key* and *secret key*
- 1) Add your API key to the params (*limit=10&ak=your-api-key*)
  - 2) Add the current timestamp to the params (*limit=10&ak=your-api-key&timestamp=1234567890*)
  - 3) Sort the query string params (*ak=your-api-key&limit=10&timestamp=1234567890*)
  - 4) Merge path and the sorted query string to a single string (*/export/categ/123.json?ak=your-api-key&limit=10&timestamp=1234567890*)
  - 5) Create a HMAC-SHA1 signature of this string using your *secret key* as the key.
  - 6) Append the hex-encoded signature to your query string: *?ak=your-api-key&limit=10&timestamp=1234567890&signature=your-signature*

Note that a signed request might be valid only for a few seconds or minutes, so you **need** to sign it right before sending it and not store the generated URL as it is likely to expire soon.

You can find example code for Python and PHP in the following sections.

If persistent signatures are enabled, you can also omit the timestamp. In this case the URL is valid forever. When using this feature, please make sure to use these URLs only where necessary - use timestamped URLs whenever possible.

## Request Signing for Python

A simple example in Python:

```
import hashlib
import hmac
import time

try:
    from urllib.parse import urlencode
except ImportError:
    from urllib import urlencode

def build_indico_request(path, params, api_key=None, secret_key=None, only_public=False, persistent=False):
    items = list(params.items()) if hasattr(params, 'items') else list(params)
    if api_key:
        items.append(('apikey', api_key))
    if only_public:
        items.append(('onlypublic', 'yes'))
    if secret_key:
        if not persistent:
            items.append(('timestamp', str(int(time.time()))))
        items = sorted(items, key=lambda x: x[0].lower())
        url = '%s?%s' % (path, urlencode(items))
        signature = hmac.new(secret_key.encode('utf-8'), url.encode('utf-8'),
                             hashlib.sha1).hexdigest()
        items.append(('signature', signature))
    if not items:
        return path
    return '%s?%s' % (path, urlencode(items))

if __name__ == '__main__':
```

(continues on next page)

(continued from previous page)

```

API_KEY = '00000000-0000-0000-0000-000000000000'
SECRET_KEY = '00000000-0000-0000-0000-000000000000'
PATH = '/export/categ/1337.json'
PARAMS = {
    'limit': 123
}
print(build_indico_request(PATH, PARAMS, API_KEY, SECRET_KEY))

```

## Request Signing for PHP

A simple example in PHP:

```

<?php

function build_indico_request($path, $params, $api_key = null, $secret_key = null,
    ↪$only_public = false, $persistent = false) {
    if($api_key) {
        $params['apikey'] = $api_key;
    }

    if($only_public) {
        $params['onlypublic'] = 'yes';
    }

    if($secret_key) {
        if(!$persistent) {
            $params['timestamp'] = time();
        }
        uksort($params, 'strcasecmp');
        $url = $path . '?' . http_build_query($params);
        $params['signature'] = hash_hmac('sha1', $url, $secret_key);
    }

    if(!$params) {
        return $path;
    }

    return $path . '?' . http_build_query($params);
}

if(true) { // change to false if you want to include this file
    $API_KEY = '00000000-0000-0000-0000-000000000000';
    $SECRET_KEY = '00000000-0000-0000-0000-000000000000';
    $PATH = '/export/categ/1337.json';
    $PARAMS = array(
        'limit' => 123
    );
    echo build_indico_request($PATH, $PARAMS, $API_KEY, $SECRET_KEY) . "\n";
}

```

### 6.1.2 Common Parameters

The following parameters are valid for all requests no matter which element is requested. If a parameter has a shorter form, it's given in parentheses.

Param	Short	Description
from/to	f/t	<b>Accepted formats:</b> <ul style="list-style-type: none"> <li>• ISO 8601 subset - YYYY-MM-DD[THH:MM]</li> <li>• ‘today’, ‘yesterday’, ‘tomorrow’ and ‘now’</li> <li>• days in the future/past: ‘[+/-]DdHHhMMm’</li> </ul>
pretty	p	Pretty-print the output. When exporting as JSON it will include whitespace to make the json more human-readable.
onlypublic	op	Only return results visible to unauthenticated users when set to <i>yes</i> .
onlyauthed	oa	Fail if the request is unauthenticated for any reason when this is set to <i>yes</i> .
cookieauth	ca	Use the Indico session cookie to authenticate instead of an API key.
nocache	nc	Disable caching of results when this is set to <i>yes</i> .
limit	n	Return no more than the X results.
offset	O	Skip the first X results.
detail	d	Specify the detail level (values depend on the exported element)
order	o	Sort the results. Must be one of <i>id</i> , <i>start</i> , <i>end</i> , <i>title</i> .
descending	c	Sort the results in descending order when set to <i>yes</i> .
tz	-	Assume given timezone (default UTC) for specified dates. Example: Europe/Lisbon.

### 6.1.3 API Resources

#### Categories

#### URL Format

*/export/categ/ID.TYPE*

The ID can be either a single category ID or a - separated list. In an authenticated request the special ID *favorites* will be resolved to the user’s list of favorites.

## Parameters

Param	Short	Description
location	l	Only include events taking place at the specified location. The * and ? wildcards may be used.
room	r	Only include events taking place in the specified room. The * and ? wildcards may be used.
type	T	Only include events of the specified type. Must be one of: simple_event (or lecture), meeting, conference

## Detail Levels

### events

Returns basic data about the events in the category.

This is the result of the following the query <https://my.indico/export/categ/2.json?from=today&to=today&pretty=yes>:

```
{
  "count": 2,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://my.indico/export/categ/2.json?from=today&to=today&pretty=yes",
  "ts": 1308841641,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-17",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-30",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Test EPayment",
      "url": "http://pcituds07.cern.ch/indico/conferenceDisplay.py?confId=137344",
      "location": "CERN",
      "_fossil": "conferenceMetadata",
      "timezone": "Europe/Zurich",
      "type": "conference",
      "id": "137344",
      "room": "1-1-025",
      "keywords": []
    },
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
```

(continues on next page)

(continued from previous page)

```

        "time": "08:00:00"
    },
    "_type": "Conference",
    "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
    },
    "description": "",
    "title": "Export Test",
    "url": "http://pcituds07.cern.ch/indico/conferenceDisplay.py?confId=137346",
    "location": "CERN",
    "_fossil": "conferenceMetadata",
    "timezone": "Europe/Zurich",
    "type": "meeting",
    "id": "137346",
    "room": null,
    "keywords": []
}
]
}

```

## Events

### URL Format

*/export/event/ID.TYPE*

The ID can be either a single event ID or a - separated list.

### Parameters

Param	Short	Description
occurrences	occ	Include the daily event times in the exported data.

## Detail Levels

### events

Returns basic data about the event. In this example occurrences are included, too.

Result for <https://indico.server/export/event/137346.json?occ=yes&pretty=yes>:

```

{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?occ=yes&pretty=yes",
  "ts": 1308899256,
  "results": [

```

(continues on next page)



(continued from previous page)

```

{
  "category": "TEST Category",
  "startDate": {
    "date": "2011-06-23",
    "tz": "Europe/Zurich",
    "time": "08:00:00"
  },
  "_type": "Conference",
  "endDate": {
    "date": "2011-06-24",
    "tz": "Europe/Zurich",
    "time": "18:00:00"
  },
  "description": "",
  "title": "Export Test",
  "url": "http://indico.server/conferenceDisplay.py?confId=137346",
  "room": null,
  "keywords": [],
  "occurrences": [
    {
      "_fossil": "period",
      "endDT": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:40:00"
      },
      "startDT": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Period"
    },
    {
      "_fossil": "period",
      "endDT": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "15:00:00"
      },
      "startDT": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "12:00:00"
      },
      "_type": "Period"
    }
  ],
  "_fossil": "conferenceMetadata",
  "timezone": "Europe/Zurich",
  "type": "meeting",
  "id": "137346",
  "location": "CERN"
}
]
}

```

## contributions

Includes the contributions of the event.

Output for <https://indico.server/export/event/137346.json?detail=contributions&pretty=yes>:

```
{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?detail=contributions&
→pretty=yes",
  "ts": 1308899252,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Export Test",
      "url": "http://indico.server/conferenceDisplay.py?confId=137346",
      "type": "meeting",
      "location": "CERN",
      "_fossil": "conferenceMetadataWithContribs",
      "timezone": "Europe/Zurich",
      "keywords": [],
      "contributions": [
        {
          "startDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:20:00"
          },
          "_type": "Contribution",
          "endDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:40:00"
          },
          "description": "",
          "title": "dlc2",
          "track": null,
          "duration": 20,
          "session": null,
          "location": "CERN",
          "_fossil": "contributionMetadata",
          "type": null,
          "id": "1",
          "room": null
        }
      ]
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    },
    {
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Contribution",
      "endDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:20:00"
      },
      "description": "",
      "title": "d1c1",
      "track": null,
      "duration": 20,
      "session": null,
      "location": "CERN",
      "_fossil": "contributionMetadata",
      "type": null,
      "id": "0",
      "room": null
    },
    {
      "startDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "14:00:00"
      },
      "_type": "Contribution",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "14:20:00"
      },
      "description": "",
      "title": "d2s1c1",
      "track": null,
      "duration": 20,
      "session": "d2s1",
      "location": "CERN",
      "_fossil": "contributionMetadata",
      "type": null,
      "id": "3",
      "room": null
    },
    {
      "startDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "12:00:00"
      },
      "_type": "Contribution",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",

```

(continues on next page)

(continued from previous page)

```

        "time": "14:00:00"
      },
      "description": "",
      "title": "d2c1",
      "track": null,
      "duration": 120,
      "session": null,
      "location": "CERN",
      "_fossil": "contributionMetadata",
      "type": null,
      "id": "2",
      "room": null
    }
  ],
  "id": "137346",
  "room": null
}

```

## subcontributions

Like *contributions*, but inside the contributions the subcontributions are included in a field named *subContributions*.

## sessions

Includes details about the different sessions and groups contributions by sessions. The top-level *contributions* list only contains contributions which are not assigned to any session. Subcontributions are included in this details level, too.

For example, <https://indico.server/export/event/137346.json?detail=sessions&pretty=yes>:

```

{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?detail=sessions&pretty=yes",
  "ts": 1308899771,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Export Test",

```

(continues on next page)

(continued from previous page)

```

"url": "http://indico.server/conferenceDisplay.py?confId=137346",
"keywords": [],
"contributions": [
  {
    "startDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:20:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:40:00"
    },
    "description": "",
    "subContributions": [],
    "title": "dlc2",
    "track": null,
    "duration": 20,
    "session": null,
    "location": "CERN",
    "_fossil": "contributionMetadataWithSubContribs",
    "type": null,
    "id": "1",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:00:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:20:00"
    },
    "description": "",
    "subContributions": [],
    "title": "dlc1",
    "track": null,
    "duration": 20,
    "session": null,
    "location": "CERN",
    "_fossil": "contributionMetadataWithSubContribs",
    "type": null,
    "id": "0",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "12:00:00"
    },

```

(continues on next page)

(continued from previous page)

```

        "_type": "Contribution",
        "endDate": {
            "date": "2011-06-24",
            "tz": "Europe/Zurich",
            "time": "14:00:00"
        },
        "description": "",
        "subContributions": [],
        "title": "d2c1",
        "track": null,
        "duration": 120,
        "session": null,
        "location": "CERN",
        "_fossil": "contributionMetadataWithSubContribs",
        "type": null,
        "id": "2",
        "room": null
    }
],
"sessions": [
    {
        "startDate": {
            "date": "2011-06-24",
            "tz": "Europe/Zurich",
            "time": "14:00:00"
        },
        "_type": "Session",
        "room": "",
        "numSlots": 1,
        "color": "#EEE0EF",
        "material": [],
        "isPoster": false,
        "sessionConveners": [],
        "location": "CERN",
        "address": "",
        "_fossil": "sessionMetadata",
        "title": "d2s1",
        "textColor": "#1D041F",
        "contributions": [
            {
                "startDate": {
                    "date": "2011-06-24",
                    "tz": "Europe/Zurich",
                    "time": "14:00:00"
                },
                "_type": "Contribution",
                "endDate": {
                    "date": "2011-06-24",
                    "tz": "Europe/Zurich",
                    "time": "14:20:00"
                },
                "description": "",
                "subContributions": [],
                "title": "d2s1c1",
                "track": null,
                "duration": 20,
                "session": "d2s1",
            }
        ]
    }
]

```

(continues on next page)

(continued from previous page)

```

        "location": "CERN",
        "_fossil": "contributionMetadataWithSubContribs",
        "type": null,
        "id": "3",
        "room": null
    }
],
    "id": "0"
}
],
    "location": "CERN",
    "_fossil": "conferenceMetadataWithSessions",
    "timezone": "Europe/Zurich",
    "type": "meeting",
    "id": "137346",
    "room": null
}
]
}

```

## Timetable

### URL Format

*/export/timetable/ID.TYPE*

The ID should be the event ID, e.g. 123.

## Results

Returns the timetable of the event.

Result for <https://indico.server/export/timetable/137346.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```

{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/timetable/137346.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "ts": 1367242732,
  "results": {
    "137346": {
      "20130429": {
        "c0": {
          "startDate": {
            "date": "2013-04-29",
            "tz": "Europe/Zurich",
            "time": "16:00:00"
          },
          "_type": "ContribSchEntry",
          "material": [],

```

(continues on next page)

(continued from previous page)

```

        "endDate": {
            "date": "2013-04-29",
            "tz": "Europe\Zurich",
            "time": "16:30:00"
        },
        "description": "",
        "title": "Contrib 1",
        "id": "c0",
        "contributionId": "0",
        "sessionSlotId": null,
        "conferenceId": "137346",
        "presenters": [],
        "sessionId": null,
        "location": "CERN",
        "uniqueId": "a137346t0",
        "_fossil": "contribSchEntryDisplay",
        "sessionCode": null,
        "entryType": "Contribution",
        "room": "160-1-009"
    }
}

```

## Event Search

### URL Format

*/export/event/search/TERM.TYPE*

The TERM should be a string, e.g. “ic hep”

### Results

Returns the events found.

Result for <https://indico.server/export/event/search/ic hep.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```

{
    "count": 5,
    "additionalInfo": {},
    "_type": "HTTPAPIResult",
    "complete": true,
    "url": "https://indico.server/export/event/search/ic hep.json?ak=00000000-
    ↪0000-0000-0000-000000000000&pretty=yes",
    "ts": 1367245058,
    "results": [
        {
            "startDate": {
                "date": "2010-07-16",
                "tz": "UTC",
                "time": "11:00:00"
            },
            "hasAnyProtection": false,

```

(continues on next page)



(continued from previous page)

```

        "id": "101465",
        "title": "Rehearsals for ICHEP Friday 16th July Afternoon Session"
    },
    {
        "startDate": {
            "date": "2010-08-06",
            "tz": "UTC",
            "time": "12:00:00"
        },
        "hasAnyProtection": false,
        "id": "102669",
        "title": "Overview of LHC physics results at ICHEP"
    },
    {
        "startDate": {
            "date": "2010-08-18",
            "tz": "UTC",
            "time": "17:00:00"
        },
        "hasAnyProtection": false,
        "id": "104128",
        "title": "Seminer Oturumu: \"ATLAS status and highlights as of ICHEP\" Dr.
↪Tayfun Ince (Universitaet Bonn)"
    },
    {
        "startDate": {
            "date": "2011-07-23",
            "tz": "UTC",
            "time": "11:00:00"
        },
        "hasAnyProtection": false,
        "id": "145521",
        "title": "89th Plenary ECFA and Joint EPS/ICHEP-ECFA Session - Grenoble,
↪France"
    },
    {
        "startDate": {
            "date": "2012-01-12",
            "tz": "UTC",
            "time": "08:00:00"
        },
        "hasAnyProtection": false,
        "id": "168897",
        "title": "ICHEP 2012 Outreach Planning Meeting"
    }
]
}

```

## Files

### General Information

The file export is only available for authenticated users, i.e. when using an API key and a signature (if enabled).

## URL Format

*/export/event/EVENT\_ID/session/SESSION\_ID/contrib/CONTRIBUTION\_ID/subcontrib/SUBCONTRIBUTION\_ID/material/MATERIAL\_ID*

All ID's should be single ID, not separated list.

The *EVENT\_ID* should be the event ID, e.g. *123*.

The *SESSION\_ID* (*optional*) should be the session ID, e.g. *4*.

The *CONTRIBUTION\_ID* (*optional*) should be the contribution ID, e.g. *3*.

The *SUBCONTRIBUTION\_ID* (*optional*) should be the sub-contribution ID, e.g. *1*.

The *MATERIAL\_ID* should be the material name if it came default group e.g. *Slides* or material ID if not, e.g. *2*.

The *RESOURCE\_ID* should be the resource ID.

Only supported *TYPE* for files is *bin* (binary data).

## Parameters

None

## Detail Levels

### file

Returns file (or an error in *JSON* format).

For example: <https://indico.server/export/event/23/session/0/contrib/3/material/slides/3.bin?ak=00000000-0000-0000-0000-000000000000>

## User

### General Information

The user export is only available for authenticated users, i.e. when using an API key and a signature (if enabled).

## URL Format

*/export/user/USER\_ID.TYPE*

The *USER\_ID* should be the user ID, e.g. *44*.

## Parameters

None

## Results

Returns the user information (or an error in *JSON* format).

Result for `https://indico.server/export/user/6.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes`:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult"
  "ts": 1610536660,
  "url": "https:\\\\indico.server\\export\\user\\6.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "results": [{
    "id": 6,
    "first_name": "Guinea",
    "last_name": "Pig",
    "full_name": "Guinea Pig"
    "email": "test@cern.ch",
    "affiliation": "CERN",
    "phone": "",
    "avatar_url": "\\user\\6\\picture-default",
    "identifier": "User:6",
  }],
}
```

## Room Booking

### Bookings

#### Creating bookings

#### General Information

The Room Booking API is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for this API, too. The request will fail if there is a collision with another booking, blocking or unavailable period.

Note that it is not possible to pre-book a room through this api.

### URL Format

`/api/roomBooking/bookRoom.TYPE`

*TYPE* should be *json* or *xml*.

### Parameters

The following parameters are required:

Param	Values	Description
location	text	Room location, e.g. <i>CERN</i>
roomid	text	Room id
from/to	f/t	<b>Start/End time for a booking. Accepted formats:</b> <ul style="list-style-type: none"> <li>• ISO 8601 subset - YYYY-MM-DD[THH:MM]</li> <li>• ‘today’, ‘yesterday’, ‘tomorrow’ and ‘now’</li> <li>• days in the future/past: ‘[+/-]DdHHhMMm’</li> </ul>
reason	text	Reason for booking a room
username	text	User login name for whom the booking will be created

## Booking a room

### POST request

Returns *reservation id* if the booking was successful or error information if there were any problems.

For example:

```
curl --data "username=jdoe&from=2012-12-30T21:30&to=2012-12-30T22:15&reason=meeting&location=CERN&roomid=189" 'http://indico.server/indico/api/roomBooking/bookRoom.json'
```

Result:

```
{
  {
    "url": "\api\roomBooking\bookRoom.json",
    "_type": "HTTPAPIResult",
    "results": {
      "reservationID": 45937
    },
    "ts": 1354695663
  }
}
```

## Retrieving bookings

### General Information

The reservation export is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for the reservation export API, too.

Please note that the room export with the *reservations* detail level is much more appropriate if you need reservations for specific rooms.

## URL Format

*/export/reservation/LOCATION.TYPE*

The *LOCATION* should be the room location, e.g. *CERN*. A - separated list of multiple locations is allowed, too.

## Parameters

Param	Short	Values	Description
occurrences	occ	yes, no	Include all occurrences of room reservations.
cancelled	cxl	yes, no	If specified only include cancelled ( <i>yes</i> ) or non-cancelled ( <i>no</i> ) reservations.
rejected	rej	yes, no	If specified only include rejected/non-rejected resvs.
confirmed	-	yes, no, pending	If specified only include bookings/pre-bookings with the given state.
archival	arch	yes, no	If specified only include bookings (not) from the past.
recurring	rec	yes, no	If specified only include bookings which are (not) recurring.
repeating	rep	yes, no	Alias for <i>recurring</i>
booked-for	bf	text (wildcards)	Only include bookings where the <i>booked for</i> field matches the given wildcard string.
occurs	-	yyyy-mm-dd	Only include bookings which have a valid occurrence on the given date. Multiple dates can be separated by commas.

## Detail Levels

### reservations

Returns detailed data about the reservations and the most important information about the booked room.

For example, <https://indico.server/export/reservation/CERN.json?ak=00000000-0000-0000-0000-000000000000&detail=reservation&from=today&to=today&pretty=yes>:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/reservation/CERN.json?ak=00000000-0000-0000-0000-000000000000&
↪detail=reservation&from=today&to=today&pretty=yes",
  "results": [
    {
      "_type": "Reservation",
      "repeat_unit": 1,
      "endDT": {
        "date": "2014-08-14",
        "tz": "Europe/Zurich",
        "time": "12:30:00"
      },
      "room": {
        "_type": "Room",
```

(continues on next page)

(continued from previous page)

```
        "fullName": "500-1-001 - Main Auditorium",
        "id": 57
    },
    "isConfirmed": true,
    "isValid": true,
    "repeatability": "daily",
    "repeat_step": 1,
    "vcList": [],
    "reason": "Summer Student Lecture programme",
    "bookedForName": "DOE, John",
    "is_rejected": false,
    "is_cancelled": false,
    "startDT": {
        "date": "2014-07-02",
        "tz": "Europe/Zurich",
        "time": "08:30:00"
    },
    "id": 63779,
    "bookingUrl": "http://indico.server/rooms/booking/CERN/63779/",
    "location": "CERN"
    }
],
"ts": 1406727843
}
```

## Rooms

### General Information

The room export is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for the room export API, too.

### URL Format

*/export/room/LOCATION/ID.TYPE*

The *LOCATION* should be the room location, e.g. *CERN*. The *ID* can be either a single room ID or a - separated list.

## Parameters

Param	Short	Values	Description
occurrences	occ	yes, no	Include all occurrences of room reservations.
cancelled	cxl	yes, no	If specified only include cancelled ( <i>yes</i> ) or non-cancelled ( <i>no</i> ) reservations.
rejected	rej	yes, no	If specified only include rejected/non-rejected resvs.
confirmed	-	yes, no, pending	If specified only include bookings/pre-bookings with the given state.
archival	arch	yes, no	If specified only include bookings (not) from the past.
recurring	rec	yes, no	If specified only include bookings which are (not) recurring.
repeating	rep	yes, no	Alias for <i>recurring</i>
booked-for	bf	text (wildcards)	Only include bookings where the <i>booked for</i> field matches the given wildcard string.
occurs	-	yyyy-mm-dd	Only include bookings which have a valid occurrence on the given date. Multiple dates can be separated by commas.

## Detail Levels

### rooms

Returns basic data about the rooms.

For example, <https://indico.server/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "longitude": "6.0542704900999995",
      "vcList": [
        "Audio Conference",
        "Built-in (MCU) Bridge",
        "CERN MCU",
        "ESnet MCU",
        "EVO",
        "H323 point2point",
        "Vidyo"
      ],
      "equipment": [
        "Blackboard",

```

(continues on next page)

(continued from previous page)

```

        "Computer Projector",
        "Ethernet",
        "Microphone",
        "PC",
        "Telephone conference",
        "Video conference",
        "Webcast/Recording",
        "Wireless"
    ],
    "roomNr": "001",
    "location": "CERN",
    "latitude": "46.23141394580001",
    "fullName": "500-1-001 - Main Auditorium",
    "id": 57,
    "bookingUrl": "/indico/rooms/room/CERN/57/book"
  }
],
  "ts": 1406729635
}

```

## reservations

Returns basic data about the rooms and their reservations in the given timeframe.

Output for <https://indico.server/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&detail=reservations&from=today&to=today&pretty=yes>:

```

{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&
↪detail=reservations&from=today&to=today&pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "reservations": [
        {
          "_type": "Reservation",
          "repeat_unit": 1,
          "endDT": {
            "date": "2014-08-14",
            "tz": "Europe/Zurich",
            "time": "12:30:00"
          },
          "isConfirmed": true,
          "isValid": true,
          "repeatability": "daily",
          "repeat_step": 1,
          "vcList": [],
          "reason": "Summer Student Lecture programme",
          "bookedForName": "DOE, John",

```

(continues on next page)



(continued from previous page)

```

        "is_rejected": false,
        "is_cancelled": false,
        "startDT": {
            "date": "2014-07-02",
            "tz": "Europe/Zurich",
            "time": "08:30:00"
        },
        "id": 63779,
        "bookingUrl": "http://pcavc005.cern.ch:8000/indico/rooms/booking/
↪CERN/63779/",
        "location": "CERN"
    },
    ],
    "longitude": "6.0542704900999995",
    "vcList": [
        "Audio Conference",
        "Built-in (MCU) Bridge",
        "CERN MCU",
        "ESnet MCU",
        "EVO",
        "H323 point2point",
        "Vidyo"
    ],
    "equipment": [
        "Blackboard",
        "Computer Projector",
        "Ethernet",
        "Microphone",
        "PC",
        "Telephone conference",
        "Video conference",
        "Webcast/Recording",
        "Wireless"
    ],
    "roomNr": "001",
    "location": "CERN",
    "latitude": "46.23141394580001",
    "fullName": "500-1-001 - Main Auditorium",
    "id": 57,
    "bookingUrl": "/indico/rooms/room/CERN/57/book"
    },
    ],
    "ts": 1406731966
}

```

## Get room by room name

### General Information

The search room export is guest allowed because the room data is public (no the reservations).

### URL Format

*/export/roomName/LOCATION/ROOMNAME.TYPE*

The *LOCATION* should be the room location, e.g. *CERN*. The *ROOMNAME* is a single ROOMNAME.

## Parameters

No parameters needed.

## Results

Returns basic data about the rooms.

For example, [https://indico.server/export/roomName/CERN/Main Auditorium.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes](https://indico.server/export/roomName/CERN/Main%20Auditorium.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes):

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/roomName/CERN/Main Auditorium.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "longitude": "6.0542704900999995",
      "vcList": [
        "Audio Conference",
        "Built-in (MCU) Bridge",
        "CERN MCU",
        "ESnet MCU",
        "EVO",
        "H323 point2point",
        "Vidyo"
      ],
      "equipment": [
        "Blackboard",
        "Computer Projector",
        "Ethernet",
        "Microphone",
        "PC",
        "Telephone conference",
        "Video conference",
        "Webcast/Recording",
        "Wireless"
      ],
      "roomNr": "001",
      "location": "CERN",
      "latitude": "46.23141394580001",
      "fullName": "500-1-001 - Main Auditorium",
      "id": 57,
      "bookingUrl": "/indico/rooms/room/CERN/57/book"
    }
  ],
  "ts": 1406732578
}
```

## 6.1.4 HTTP API Tools

---

**Note:** API keys and signatures have been deprecated. Please consider using *API Token Authentication* instead.

---

Deprecated since version 3.0.



This part of the documentation focuses on the core modules of Indico and includes information about the **models** and **utility functions and classes** that are useful for understanding the internals of the application.

## 7.1 API reference

This part of the documentation focuses on the core modules of Indico and includes information about the **models** and **utility functions and classes** that are useful for understanding the internals of the application.

### 7.1.1 Event

---

**Todo:** Docstrings (module, models, operations, utilities, settings)

---

#### Models

```
class indico.modules.events.models.events.Event (**kwargs)
    Bases:
        indico.core.db.sqlalchemy.searchable.SearchableTitleMixin,
        indico.core.db.sqlalchemy.descriptions.DescriptionMixin,
        indico.core.db.sqlalchemy.locations.LocationMixin,
        indico.core.db.sqlalchemy.protection.ProtectionManagersMixin,
        indico.core.db.sqlalchemy.attachments.AttachedItemsMixin,
        indico.core.db.sqlalchemy.notes.AttachedNotesMixin,
        indico.modules.events.models.persons.PersonLinkDataMixin,
        sqlalchemy.orm.decl_api.Model
```

An Indico event.

This model contains the most basic information related to an event.

Note that the ACL is currently only used for managers but not for view access!

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**ATTACHMENT\_FOLDER\_ID\_COLUMN** = 'event\_id'

**access\_key**

**acl\_entries**

The ACL entries for the event

**additional\_info**

**all\_notes**

**allow\_access\_key** = True

**allow\_location\_inheritance** = False

**allow\_no\_access\_contact** = True

**can\_display** (*user*)

Check whether the user can display the event in the category.

**can\_lock** (*user*)

Check whether the user can lock/unlock the event.

**category**

The category containing the event

**classmethod category\_chain\_overlaps** (*category\_ids*)

Create a filter that checks whether the event has any of the provided category ids in its parent chain.

Warning: This method cannot be used in a negated filter.

**Parameters** **category\_ids** – A list of category ids or a single category id

**category\_id**

The ID of immediate parent category of the event

**cfa**

**cfp**

**cloned\_from**

The event this one was cloned from

**cloned\_from\_id**

If this event was cloned, the id of the parent event

**contact\_emails**

**contact\_phones**

**contact\_title**

**created\_dt**

The creation date of the event

**creator**

The user who created the event

**creator\_id**

The ID of the user who created the event

**custom\_boa**

The custom book of abstracts

**custom\_boa\_id**

The ID of the uploaded custom book of abstracts (if available)

**default\_page**

The event's default page (conferences only)

**default\_page\_id**

The ID of the event's default page (conferences only)

**default\_render\_mode = 1****delete** (*reason*, *user=None*)**disallowed\_protection\_modes = frozenset()****display\_tzinfo**

The tzinfo of the event as preferred by the current user.

**duration****editable\_types****end\_dt**

The end date of the event

**end\_dt\_display**

The 'displayed end dt', which is usually the actual end dt, but may be overridden for a conference.

**end\_dt\_local****end\_dt\_override****ends\_after** (*dt*)

Check whether the event ends on/after the specified date.

**event**

Convenience property so all event entities have it.

**external\_logo\_url****external\_url**
**get\_allowed\_sender\_emails** (*include\_current\_user=True*, *include\_creator=True*, *include\_managers=True*, *include\_contact=True*, *include\_chairs=True*, *extra=None*)

Return the emails of people who can be used as senders (or rather Reply-to contacts) in emails sent from within an event.

**Parameters**

- **include\_current\_user** – Whether to include the email of the currently logged-in user
- **include\_creator** – Whether to include the email of the event creator
- **include\_managers** – Whether to include the email of all event managers
- **include\_contact** – Whether to include the “event contact” emails
- **include\_chairs** – Whether to include the emails of event chairpersons (or lecture speakers)
- **extra** – An email address that is always included, even if it is not in any of the included lists.

**Returns** A dictionary mapping emails to pretty names

**get\_contribution** (*id\_*)

Get a contribution of the event.

**get\_contribution\_field** (*field\_id*)

**get\_label\_markup** (*size*=")

**get\_non\_inheriting\_objects** ()

Get a set of child objects that do not inherit protection.

**get\_relative\_event\_ids** ()

Get the first, last, previous and next event IDs.

Any of those values may be `None` if there is no matching event or if it would be the current event.

**Returns** A dict containing `first`, `last`, `prev` and `next`.

**get\_session** (*id\_=None*, *friendly\_id=None*)

Get a session of the event.

**get\_session\_block** (*id\_*, *scheduled\_only=False*)

Get a session block of the event.

**get\_sorted\_tracks** ()

Return tracks and track groups in the correct order.

**get\_verbose\_title** (*show\_speakers=False*, *show\_series\_pos=False*)

Get the event title with some additional information.

#### Parameters

- **show\_speakers** – Whether to prefix the title with the speakers of the event.
- **show\_series\_pos** – Whether to suffix the title with the position and total count in the event's series.

**happens\_between** (*from\_dt=None*, *to\_dt=None*)

Check whether the event takes place within two dates.

**has\_custom\_boa**

**has\_ended**

**has\_feature** (*feature*)

Check if a feature is enabled for the event.

**has\_logo**

**has\_regform\_in\_acl**

**has\_stylesheet**

**id**

The ID of the event

**inherit\_location** = **False**

**inheriting\_have\_acl** = **True**

**is\_deleted**

If the event has been deleted

**is\_locked**

If the event is locked (read-only mode)



**is\_user\_registered** (*user*)

Check whether the user is registered in the event.

This takes both unpaid and complete registrations into account.

**classmethod is\_visible\_in** (*category\_id*)

Create a filter that checks whether the event is visible in the specified category.

**iter\_days** (*tzinfo=None*)

**keywords**

A list of tags/keywords for the event

**label**

The label assigned to the event

**label\_id**

The ID of the label assigned to the event

**label\_message**

**location\_backref\_name** = 'events'

**locator**

**log** (*realm, kind, module, summary, user=None, type\_='simple', data=None, meta=None*)

Create a new log entry for the event.

#### Parameters

- **realm** – A value from *EventLogRealm* indicating the realm of the action.
- **kind** – A value from *EventLogKind* indicating the kind of the action that was performed.
- **module** – A human-friendly string describing the module related to the action.
- **summary** – A one-line summary describing the logged action.
- **user** – The user who performed the action.
- **type** – The type of the log entry. This is used for custom rendering of the log message/data
- **data** – JSON-serializable data specific to the log type.
- **meta** – JSON-serializable data that won't be displayed.

**Returns** The newly created *EventLogEntry*

In most cases the `simple` log type is fine. For this type, any items from `data` will be shown in the detailed view of the log entry. You may either use a dict (which will be sorted) alphabetically or a list of `key, value` pairs which will be displayed in the given order.

**log\_entries**

**logging\_disabled**

Temporarily disable event logging.

This is useful when performing actions e.g. during event creation or at other times where adding entries to the event log doesn't make sense.

**logo**

The logo's raw image data

**logo\_metadata**

The metadata of the logo (hash, size, filename, content\_type)

**logo\_url**

**map\_url**

**move** (*category*)

**move\_start\_dt** (*start\_dt*)  
Set event start\_dt and adjust its timetable entries.

**note**

**organizer\_info**

**own\_address**

**own\_map\_url**  
The url to a map for the event

**own\_no\_access\_contact**

**own\_room**

**own\_room\_id**

**own\_room\_name**

**own\_venue**

**own\_venue\_id**

**own\_venue\_name**

**participation\_regform**

**person\_links**  
Persons associated with this event

**possible\_render\_modes** = {<RenderMode.html: 1>}

**preload\_all\_acl\_entries** ()

**protection\_mode**

**protection\_parent**  
The parent object to consult for ProtectionMode.inheriting.

**public\_regform\_access**

**published\_registrations**

**references**  
External references associated with this event

**refresh\_event\_persons** (\*, *notify=True*)  
Update the data for all EventPersons based on the linked Users.  
  
Parameters **notify** – Whether to trigger the person\_updated signal.

**render\_mode** = 1

**reservations**

**restore** (*reason=None, user=None*)

**scheduled\_notes**

**series**  
The series this event is part of

```

series_id
    The ID of the series this events belongs to

session_block_count

short_external_url

short_url

start_dt
    The start date of the event

start_dt_display
    The 'displayed start dt', which is usually the actual start dt, but may be overridden for a conference.

start_dt_local

start_dt_override

starts_between (from_dt=None, to_dt=None)
    Check whether the event starts within two dates.

stylesheet
    The stylesheet's raw image data

stylesheet_metadata
    The metadata of the stylesheet (hash, size, filename)

theme

timetable_entries

timezone
    The timezone of the event

title

type

type_

tzinfo

url

url_shortcut
    The URL shortcut for the event

visibility
    The visibility depth in category overviews

class indico.modules.events.models.events.EventType
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    conference = 3

    lecture = 1

    legacy_name

    meeting = 2

class indico.modules.events.models.persons.AuthorsSpeakersMixin
    Bases: object

    AUTHORS_SPEAKERS_DISPLAY_ORDER_ATTR = 'display_order_key'

```

**primary\_authors**

**secondary\_authors**

**speakers**

```
class indico.modules.events.models.persons.EventPerson (**kwargs)
    Bases:      indico.modules.users.models.users.PersonMixin,      sqlalchemy.orm.
    decl_api.Model
```

A person inside an event, e.g. a speaker/author etc.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**address**

**affiliation**

**classmethod** **create\_from\_user** (*user*, *event=None*, *is\_untrusted=False*)

**email**

**event**

**event\_id**

**first\_name**

**classmethod** **for\_user** (*user*, *event=None*, *is\_untrusted=False*)

Return EventPerson for a matching User in Event creating if needed.

**has\_role** (*role*, *obj*)

Whether the person has a role in the ACL list of a given object.

**id**

**identifier**

**invited\_dt**

**is\_untrusted**

**last\_name**

**classmethod** **link\_user\_by\_email** (*user*)

Link all email-based persons matching the user's email addresses with the user.

**Parameters** **user** – A User object.

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**merge\_person\_info** (*other*)

**classmethod merge\_users** (*target, source*)

Merge the EventPersons of two users.

#### Parameters

- **target** – The target user of the merge
- **source** – The user that is being merged into *target*

**phone**

**principal**

**sync\_user** (\*, *notify=True*)

Update all person data based on the current user data.

**Parameters notify** – Whether to trigger the `person_updated` signal.

**user**

**user\_id**

**class** `indico.modules.events.models.persons.EventPersonLink` (\*args, \*\*kwargs)

Bases: `indico.modules.events.models.persons.PersonLinkBase`

Association between EventPerson and Event.

Chairperson or speaker (lecture)

**display\_order**

**event\_id**

**id**

**is\_submitter**

**object\_relationship\_name** = 'event'

**person**

**person\_id**

**person\_link\_backref\_name** = 'event\_links'

**person\_link\_unique\_columns** = ('event\_id',)

**class** `indico.modules.events.models.persons.PersonLinkBase` (\*args, \*\*kwargs)

Bases: `indico.modules.users.models.users.PersonMixin`, `sqlalchemy.orm.`

`decl_api.Model`

Base class for EventPerson associations.

**address**

```
affiliation
display_order = Column(None, Integer(), table=None, nullable=False, default=ColumnDefault(0))
display_order_key
display_order_key_lastname
email
first_name
id = Column(None, Integer(), table=None, primary_key=True, nullable=False)
last_name
object
object_relationship_name = None
    The name of the relationship pointing to the object the person is linked to
person = <RelationshipProperty at 0x7fbaff5e4cc0; no key>
person_id = Column(None, Integer(), ForeignKey('events.persons.id'), table=None, nullable=True)
person_link_backref_name = None
    The name of the backref on the EventPerson
person_link_unique_columns = None
    The columns which should be included in the unique constraint.
phone
title
```

```
class indico.modules.events.models.persons.PersonLinkDataMixin
    Bases: object

    person_link_data
```

```
class indico.modules.events.models.principals.EventPrincipal(**kwargs)
    Bases: indico.core.db.sqlalchemy.principals.PrincipalPermissionsMixin,
           sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    allow_category_roles = True
    allow_emails = True
    allow_event_roles = True
    allow_networks = True
    allow_registration_forms = True
    category_role
    category_role_id
    email
    event_id
    The ID of the associated event
```

```

event_role
event_role_id
full_access
id
    The ID of the acl entry
ip_network_group
ip_network_group_id
local_group
local_group_id
multipass_group_name
multipass_group_provider
permissions
principal_backref_name = 'in_event_acls'
principal_for = 'Event'
read_access
registration_form
registration_form_id
type
unique_columns = ('event_id',)
user
user_id

```

```
class indico.modules.events.models.references.EventReference (**kwargs)
```

Bases: *indico.modules.events.models.references.ReferenceModelBase*

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

event_id
id
reference_backref_name = 'event_references'
reference_type
reference_type_id
value

```

```
class indico.modules.events.models.references.ReferenceModelBase (**kwargs)
```

Bases: *sqlalchemy.orm.decl\_api.Model*

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id** = `Column(None, Integer(), table=None, primary_key=True, nullable=False)`

**reference\_backref\_name** = `None`

The name of the backref on the *ReferenceType*

**reference\_type** = `<RelationshipProperty at 0x7fbaff29ee40; no key>`

**reference\_type\_id** = `Column(None, Integer(), ForeignKey('indico.reference_types.id'), t`

**url**

The URL of the referenced entity.

`None` if no URL template is defined.

**urn**

The URN of the referenced entity.

`None` if no scheme is defined.

**value** = `Column(None, String(), table=None, nullable=False)`

**class** `indico.modules.events.models.references.ReferenceType(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

The unique ID of the reference type

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**name**

The name of the referenced system

**scheme**

The scheme used to build an URN for the reference



**url\_template**

A URL template to build a link to a referenced entity

```
class indico.modules.events.models.reviews.ProposalCommentMixin
```

Bases: `object`

**can\_edit** (*user*)

**timeline\_item\_type** = `'comment'`

```
class indico.modules.events.models.reviews.ProposalGroupProxy (group)
```

Bases: `object`

The object that the proposals can be grouped by.

It provides all necessary methods for building the URLs, displaying the grouping information, etc.

**full\_title**

**full\_title\_attr** = `'full_title'`

**id****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**title**

**title\_attr** = `'title'`

```
class indico.modules.events.models.reviews.ProposalMixin
```

Bases: `object`

Classes that represent a proposal object should extend this class (ex: Abstract, Paper).

**call\_for\_proposals\_attr** = `None`

Attribute to retrieve the object with access to the reviewing settings

**can\_comment** (*user*)

**can\_review** (*user*, *check\_state=False*)

**cfp**

**create\_comment\_endpoint** = `None`

**create\_judgment\_endpoint** = `None`

```
create_review_endpoint = None
delete_comment_endpoint = None
edit_comment_endpoint = None
edit_review_endpoint = None
get_delete_comment_url (comment)
get_last_revision ()
get_revisions ()
get_save_comment_url (comment=None)
get_save_judgment_url ()
get_save_review_url (group=None, review=None)
is_in_final_state
proposal_type = None
    A unique identifier to handle rendering differences between proposal types
revisions_enabled = True
    Whether there is support for multiple revisions per proposal or just one
```

**class** `indico.modules.events.models.reviews.ProposalReviewMixin`  
Bases: `object`

Mixin for proposal reviews.

Classes that represent a review of a proposal should extend this class (ex: `AbstractReview`, `PaperReview`).

```
can_edit (user)

group

group_attr = None
    Object used to group reviews together

group_proxy_cls
    Proxy class to provide the necessary properties and methods to the review grouping object
    alias of ProposalGroupProxy

revision

revision_attr = None
    The revision object that the review refers to

score

timeline_item_type = 'review'
    A unique identifier to handle rendering differences between timeline items
```

**class** `indico.modules.events.models.reviews.ProposalRevisionMixin`  
Bases: `object`

Properties and methods of a proposal revision.

```
get_reviewed_for_groups (user, include_reviewed=False)
get_reviewer_render_data (user)
get_reviews (group=None, user=None)
get_timeline (user=None)
```

**proposal**

**proposal\_attr = None**

The attribute of the revision used to fetch the proposal object.

**revisions\_enabled = True**

Whether the reviewing process supports multiple revisions per proposal. If set to false it is assumed that the reviewing process supports only one revision per proposal.

**class** `indico.modules.events.models.series.EventSeries` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A series of events.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

The ID of the series

**show\_links**

Whether to show links to the other events in the same series on the main event page.

**show\_sequence\_in\_title**

Whether to show the sequence number of an event in its title on category display pages and on the main event page.

**class** `indico.modules.events.models.settings.EventSetting` (*\*\*kwargs*)

Bases: `indico.core.settings.models.base.JSONSettingsBase`, `indico.modules.events.models.settings.EventSettingsMixin`, `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**event**

**event\_id**

**id**

**module**

**name**

**settings\_backref\_name = 'settings'**

**value**

**class** `indico.modules.events.models.settings.EventSettingPrincipal` (*\*\*kwargs*)

Bases: `indico.core.settings.models.base.PrincipalSettingsBase`, `indico.modules.events.models.settings.EventSettingsMixin`, `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_category_roles = True
allow_event_roles = True
category_role
category_role_id
email = None
event
event_id
event_role
event_role_id
extra_key_cols = ('event_id',)
id
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
module
multipass_group_name
multipass_group_provider
name
principal_backref_name = 'in_event_settings_acls'
registration_form = None
registration_form_id = None
settings_backref_name = 'settings_principals'
type
user
user_id
```

```
class indico.modules.events.models.settings.EventSettingsMixin
    Bases: object
```

```
    event = <RelationshipProperty at 0x7fbaff15acc0; no key>
```

```
    event_id = Column(None, Integer(), ForeignKey('events.events.id'), table=None, nullable=True)
```

```
    settings_backref_name = None
```

```
class indico.modules.events.models.static_list_links.StaticListLink(**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

Display configuration data used in static links to listing pages.

This allows users to share links to listing pages in events while preserving e.g. column/filter configurations.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**classmethod** `create(event, type_, data)`

Create a new static list link.

If one exists with the same data, that link is used instead of creating a new one.

#### Parameters

- **event** – the *Event* for which to create the link
- **type** – the type of the link
- **data** – the data to associate with the link

**Returns** the newly created *StaticListLink*

**created\_dt**

**data**

**event**

**event\_id**

**id**

**last\_used\_dt**

**classmethod** `load(event, type_, uuid)`

Load the data associated with a link.

#### Parameters

- **event** – the *Event* the link belongs to
- **type** – the type of the link
- **uuid** – the UUID of the link

**Returns** the link data or `None` if the link does not exist

**type**

**uuid**

## Operations

`indico.modules.events.operations.clone_event(event, n_occurrence, start_dt, cloners, category=None, refresh_users=False)`

Clone an event on a given date/time.

Runs all required cloners.

#### Parameters

- **n\_occurrence** – The 1-indexed number of the occurrence, if this is a “recurring” clone, otherwise 0
- **start\_dt** – The start datetime of the new event;
- **cloners** – A set containing the names of all enabled cloners;

- **category** – The *Category* the new event will be created in.

**Aparam refresh\_users** Whether *EventPerson* data should be updated from their linked *User* object

```
indico.modules.events.operations.clone_into_event (source_event, target_event, cloners)
```

Clone data into an existing event.

Runs all required cloners.

#### Parameters

- **source\_event** – The *Event* to clone data from;
- **target\_event** – The *Event* to clone data into;
- **cloners** – A set containing the names of all enabled cloners.

```
indico.modules.events.operations.create_event (category, event_type, data,
                                                add_creator_as_manager=True, features=None, cloning=False)
```

Create a new event.

#### Parameters

- **category** – The category in which to create the event
- **event\_type** – An *EventType* value
- **data** – A dict containing data used to populate the event
- **add\_creator\_as\_manager** – Whether the creator (current user) should be added as a manager
- **features** – A list of features that will be enabled for the event. If set, only those features will be used and the default feature set for the event type will be ignored.
- **cloning** – Whether the event is created via cloning or not

```
indico.modules.events.operations.create_event_label (data)
```

```
indico.modules.events.operations.create_event_references (event, data)
```

```
indico.modules.events.operations.create_reference_type (data)
```

```
indico.modules.events.operations.create_reviewing_question (event, question_model,
                                                            wtf_field_cls, form,
                                                            data=None)
```

```
indico.modules.events.operations.delete_event_label (event_label)
```

```
indico.modules.events.operations.delete_reference_type (reference_type)
```

```
indico.modules.events.operations.delete_reviewing_question (question)
```

```
indico.modules.events.operations.lock_event (event)
```

```
indico.modules.events.operations.sort_reviewing_questions (questions,
                                                            new_positions)
```

```
indico.modules.events.operations.unlock_event (event)
```

```
indico.modules.events.operations.update_event (event, update_timetable=False, **data)
```

```
indico.modules.events.operations.update_event_label (event_label, data)
```

```
indico.modules.events.operations.update_event_protection (event, data)
```

```

indico.modules.events.operations.update_event_type(event, type_)
indico.modules.events.operations.update_reference_type(reference_type, data)
indico.modules.events.operations.update_reviewing_question(question, form)

```

## Utilities

```
class indico.modules.events.util.ListGeneratorBase(event, entry_parent=None)
```

Bases: `object`

Base class for classes performing actions on Indico object lists.

### Parameters

- **event** – The associated *Event*
- **entry\_parent** – The parent of the entries of the list. If it's `None`, the parent is assumed to be the event itself.

```
default_list_config = None
```

The default list configuration dictionary

```
endpoint = None
```

The endpoint of the list management page

```
entry_parent = None
```

The parent object of the list items

```
event = None
```

The event the list is associated with

```
flash_info_message(obj)
```

```
generate_static_url()
```

Return a URL with a uuid referring to the list's configuration.

```
get_list_url(uuid=None, external=False)
```

Return the URL of the list management page.

```
list_link_type = None
```

Unique list identifier

```
static_items = None
```

Columns that originate from the list item's properties, relationships etc, but not from user defined fields (e.g. registration/contribution fields)

```
store_configuration()
```

Load the filters from the request and store them in the session.

```
class indico.modules.events.util.ZipGeneratorMixin
```

Bases: `object`

Mixin for RHs that generate zip with files.

```
indico.modules.events.util.check_event_locked(rh, event, force=False)
```

```
indico.modules.events.util.check_permissions(event, field, allow_networks=False)
```

```
indico.modules.events.util.create_event_logo_tmp_file(event, tmpdir=None)
```

Create a temporary file with the event's logo.

If *tmpdir* is specified, the logo file is created in there and a path relative to that directory is returned.

```
indico.modules.events.util.get_all_user_roles(event, user)
```

`indico.modules.events.util.get_event_from_url(url)`

`indico.modules.events.util.get_events_created_by(user, dt=None)`

Get the IDs of events created by the user.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

**Returns** A set of event ids

`indico.modules.events.util.get_events_managed_by(user, dt=None)`

Get the IDs of events where the user has management privs.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

**Returns** A set of event ids

`indico.modules.events.util.get_events_with_linked_event_persons(user, dt=None)`

Return a dict containing the event ids and role for all events where the user is a chairperson or (in case of a lecture) speaker.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

`indico.modules.events.util.get_field_values(form_data)`

Split the form fields between custom and static.

`indico.modules.events.util.get_object_from_args(args=None)`

Retrieve an event object from request arguments.

This utility is meant to be used in cases where the same controller can deal with objects attached to various parts of an event which use different URLs to indicate which object to use.

**Parameters** **args** – The request arguments. If unspecified, `request.view_args` is used.

**Returns** An `(object_type, event, object)` tuple. The event is always the *Event* associated with the object. The object may be an *Event*, *Session*, *Contribution* or *SubContribution*. If the object does not exist, `(object_type, None, None)` is returned.

`indico.modules.events.util.get_random_color(event)`

`indico.modules.events.util.get_theme(event, override_theme_id=None)`

Get the theme ID and whether it's an override.

This is useful for places where a user may specify a different timetable theme. If the override theme is not valid for the event, a message is flashed and an exception redirecting the user to the main event page is raised.

**Raises** **BadRequest** – if the override theme id is not valid

**Returns** a `(theme_id, is_override)` tuple

`indico.modules.events.util.register_event_time_change(event)`

Register a time-related change for an event.

This is an internal helper function used in the model to record changes of the start time or end time. The changes are exposed through the `track_time_changes` contextmanager function.



```
indico.modules.events.util.register_location_change(entry)
```

Register a location-related change for an event object.

This is an internal helper function used in the models to record changes of the location information. The changes are exposed through the `track_location_changes` contextmanager function.

```
indico.modules.events.util.register_time_change(entry)
```

Register a time-related change for a timetable entry.

This is an internal helper function used in the models to record changes of the start time or duration. The changes are exposed through the `track_time_changes` contextmanager function.

```
indico.modules.events.util.serialize_event_for_ical(event, detail_level)
```

```
indico.modules.events.util.serialize_event_for_json_ld(event, full=False)
```

```
indico.modules.events.util.serialize_event_person(person)
```

Serialize EventPerson to JSON-like object.

```
indico.modules.events.util.serialize_person_for_json_ld(person)
```

```
indico.modules.events.util.serialize_person_link(person_link)
```

Serialize PersonLink to JSON-like object.

```
indico.modules.events.util.set_custom_fields(obj, custom_fields_data)
```

```
indico.modules.events.util.should_show_draft_warning(event)
```

```
indico.modules.events.util.track_location_changes()
```

Track location changes of event objects.

This provides a list of changes while the context manager was active and also triggers `location_changed` signals.

If the code running inside the `with` block of this context manager raises an exception, no signals will be triggered.

```
indico.modules.events.util.track_time_changes(auto_extend=False, user=None)
```

Track time changes of event objects.

This provides a list of changes while the context manager was active and also triggers `times_changed` signals.

If the code running inside the `with` block of this context manager raises an exception, no signals will be triggered.

#### Parameters

- **auto\_extend** – Whether entry parents will get their boundaries automatically extended or not. Passing 'start' will extend only start datetime, 'end' to extend only end datetime.
- **user** – The *User* that will trigger time changes.

```
indico.modules.events.util.update_object_principals(obj, new_principals,
                                                    read_access=False,
                                                    full_access=False,    permis-
                                                    sion=None)
```

Update an object's ACL with a new list of principals.

Exactly one argument out of `read_access`, `full_access` and `role` must be specified.

#### Parameters

- **obj** – The object to update. Must have `acl_entries`
- **new\_principals** – The set containing the new principals
- **read\_access** – Whether the read access ACL should be updated

- **full\_access** – Whether the full access ACL should be updated
- **permission** – The role ACL that should be updated

## Settings

**class** `indico.modules.events.settings.EventACLProxy` (*proxy*)

Bases: `indico.core.settings.proxy.ACLProxyBase`

Proxy class for event-specific ACL settings.

**add\_principal** (*event, name, principal*)

Add a principal to an ACL.

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

**contains\_user** (*event, name, user*)

Check if a user is in an ACL.

To pass this check, the user can either be in the ACL itself or in a group in the ACL.

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **user** – A *User*

**get** (*event, name*)

Retrieves an ACL setting

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name

**merge\_users** (*target, source*)

Replace all ACL user entries for *source* with *target*.

**remove\_principal** (*event, name, principal*)

Remove a principal from an ACL.

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

**set** (*event, name, acl*)

Replace an ACL with a new one.

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name

- **acl** – A set containing principals (users/groups)

```
class indico.modules.events.settings.EventSettingProperty (proxy, name, default=<object object>, attr=None)
```

Bases: `indico.core.settings.proxy.SettingProperty`

**attr** = 'event'

```
class indico.modules.events.settings.EventSettingsProxy (module, defaults=None, strict=True, acls=None, converters=None)
```

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access event-specific settings for a certain module.

**acl\_proxy\_class**  
alias of `EventACLProxy`

**delete** (*event, \*names*)  
Delete settings.

#### Parameters

- **event** – Event (or its ID)
- **names** – One or more names of settings to delete

**delete\_all** (*event*)  
Delete all settings.

**Parameters** **event** – Event (or its ID)

**get** (*event, name, default=<object object>*)  
Retrieve the value of a single setting.

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

**Returns** The settings's value or the default value

**get\_all** (*event, no\_defaults=False*)  
Retrieve all settings.

#### Parameters

- **event** – Event (or its ID)
- **no\_defaults** – Only return existing settings and ignore defaults.

**Returns** Dict containing the settings

**query**  
Return a query object filtering by the proxy's module.

**set** (*event, name, value*)  
Set a single setting.

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name

- **value** – Setting value; must be JSON-serializable

**set\_multi** (*event*, *items*)  
Set multiple settings at once.

#### Parameters

- **event** – Event (or its ID)
- **items** – Dict containing the new settings

**class** `indico.modules.events.settings.ThemeSettingsProxy`

Bases: `object`

**defaults**

**get\_themes\_for** (*event\_type*)

**settings**

**themes**

`indico.modules.events.settings.event_or_id(f)`

## 7.1.2 Abstract

---

**Todo:** Docstrings (module, models, operations, utilities, settings)

---

### Models

**class** `indico.modules.events.abstracts.models.abstracts.Abstract` (*\*\*kwargs*)  
Bases: `indico.modules.events.models.reviews.ProposalMixin`, `indico.modules.events.models.reviews.ProposalRevisionMixin`, `indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.modules.events.contributions.models.contributions.CustomFieldsMixin`, `indico.modules.events.models.persons.AuthorsSpeakersMixin`, `sqlalchemy.orm.decl_api.Model`

An abstract that can be associated to a Contribution.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**AUTHORS\_SPEAKERS\_DISPLAY\_ORDER\_ATTR** = 'display\_order\_key\_lastname'

**accepted\_contrib\_type**

**accepted\_contrib\_type\_id**

**accepted\_track**

**accepted\_track\_id**

**call\_for\_proposals\_attr** = 'cfa'

**can\_access** (*user*)

**can\_change\_tracks** (*user*, *check\_state=False*)

```

can_comment (user, check_state=False)
can_convene (user)
can_edit (user)
can_judge (user, check_state=False)
can_review (user, check_state=False)
can_see_reviews (user)
can_withdraw (user, check_state=False)
candidate_contrib_types
candidate_tracks
create_comment_endpoint = 'abstracts.comment_abstract'
create_judgment_endpoint = 'abstracts.judge_abstract'
create_review_endpoint = 'abstracts.review_abstract'
data_by_field
default_render_mode = 2
delete_comment_endpoint = 'abstracts.delete_abstract_comment'
duplicate_of
duplicate_of_id
edit_comment_endpoint = 'abstracts.edit_abstract_comment'
edit_review_endpoint = 'abstracts.edit_review'
edit_track_mode
event
event_id
field_values
    Data stored in abstract/contribution fields
friendly_id
get_reviewed_for_groups (user, include_reviewed=False)
get_timeline (user=None)
get_track_question_scores ()
get_track_reviewing_state (track)
get_track_score (track)
id
is_deleted
is_in_final_state
judge
    User who judged the abstract
judge_id
    ID of the user who judged the abstract

```

`judgment_comment`

`judgment_dt`

`locator`

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

`log(*args, **kwargs)`

Log with prefilled metadata for the abstract.

`marshmallow_aliases = {'_description': 'content'}`

`merged_into`

`merged_into_id`

`modification_ended`

`modified_by`

`modified_by_id`

`modified_dt`

`person_links`

Persons associated with this abstract

`possible_render_modes = {<RenderMode.markdown: 2>}`

`proposal_type = 'abstract'`

`public_state`

`render_mode = 2`

`reset_state()`

`reviewed_for_tracks`

`reviewing_state`

`revisions_enabled = False`

`score`

`state`

`submission_comment`

**submitted\_contrib\_type**  
**submitted\_contrib\_type\_id**  
**submitted\_dt**  
**submitted\_for\_tracks**  
**submitter**  
User who submitted the abstract  
**submitter\_id**  
ID of the user who submitted the abstract  
**title**  
**user\_owns** (*user*)  
**uuid**  
**verbose\_title**

**class** indico.modules.events.abstracts.models.abstracts.**AbstractPublicState**  
Bases: indico.util.enum.RichIntEnum

An enumeration.

**accepted** = 3  
**awaiting** = -1  
**duplicate** = 6  
**invited** = 7  
**merged** = 5  
**rejected** = 4  
**under\_review** = -2  
**withdrawn** = 2

**class** indico.modules.events.abstracts.models.abstracts.**AbstractReviewingState**  
Bases: indico.util.enum.RichIntEnum

An enumeration.

**conflicting** = 3  
**in\_progress** = 1  
**mixed** = 5  
**negative** = 4  
**not\_started** = 0  
**positive** = 2

**class** indico.modules.events.abstracts.models.abstracts.**AbstractState**  
Bases: indico.util.enum.RichIntEnum

An enumeration.

**accepted** = 3  
**duplicate** = 6  
**invited** = 7

```
merged = 5
rejected = 4
submitted = 1
withdrawn = 2

class indico.modules.events.abstracts.models.abstracts.EditTrackMode
    Bases: int, indico.util.enum.IndicoEnum

    An enumeration.

    both = 1
    none = 0
    reviewed_for = 2

class indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts(event)
    Bases: object

    Proxy class to facilitate access to the call for abstracts settings.

    allow_attachments
    allow_comments
    allow_contributors_in_comments
    allow_convener_judgment
    allow_convener_track_change
    allow_editing
    announcement
    can_edit_abstracts(user)
    can_submit_abstracts(user)
    close()
    contribution_submitters
    end_dt
    has_ended
    has_started
    is_open
    is_scheduled
    judgment_instructions
    modification_end_dt
    modification_ended
    open()
    rating_range
    reviewing_instructions
    schedule(start_dt, end_dt, modification_end_dt)
    start_dt
```



**submission\_instructions**

```
class indico.modules.events.abstracts.models.comments.AbstractComment (**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalCommentMixin, indico.
            core.db.sqlalchemy.review_comments.ReviewCommentMixin, sqlalchemy.orm.
            decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**abstract****abstract\_id**

**can\_edit** (*user*)

**can\_view** (*user*)

**created\_dt****id****is\_deleted****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**marshmallow\_aliases** = {'\_text': 'text'}

**modified\_by****modified\_by\_id****modified\_dt**

**render\_mode** = 2

**user**

**user\_backref\_name** = 'abstract\_comments'

**user\_id**

**user\_modified\_backref\_name** = 'modified\_abstract\_comments'

**visibility**

```
class indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**abstract****abstract\_id****body**

```
classmethod create_from_email (email_data, email_tpl, user=None)
    Create a new log entry from the data used to send an email.
```

**Parameters**

- **email\_data** – email data as returned from *make\_email*
- **email\_tpl** – the abstract email template that created the email
- **user** – the user who performed the action causing the notification

**data****email\_template****email\_template\_id****id****recipients****sent\_dt****subject****user****user\_id**

```
class indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

An email template for abstracts notifications.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**body**

The body of the template

**event****event\_id****extra\_cc\_emails**

List of extra email addresses to be added as CC in the email

**id**

**include\_authors**

Whether to include authors' email addresses as To for emails

**include\_coauthors**

Whether to include co-authors' email addresses as CC for emails

**include\_submitter**

Whether to include the submitter's email address as To for emails

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**position**

The relative position of the template in the list of templates

**reply\_to\_address**

The address to use as Reply-To in the email

**rules**

Conditions need to be met to send the email

**stop\_on\_match**

Whether to stop checking the rest of the conditions when a match is found

**subject**

The subject of the email

**title**

```
class indico.modules.events.abstracts.models.fields.AbstractFieldValue (**kwargs)
    Bases: indico.modules.events.contributions.models.fields.
    ContributionFieldValueBase
```

Store a field values related to abstracts.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**abstract\_id**

**contribution\_field**

**contribution\_field\_backref\_name** = 'abstract\_values'

**contribution\_field\_id**

**data**

```
class indico.modules.events.abstracts.models.files.AbstractFile(**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**abstract**

**abstract\_id**

**add\_file\_date\_column** = False

**content\_type**

The MIME type of the file.

**created\_dt** = None

**extension**

The extension of the file.

**filename**

The name of the file.

**id**

**locator**

**md5**

An MD5 hash of the file.

Automatically assigned when *save()* is called.

**size**

The size of the file (in bytes).

Automatically assigned when *save()* is called.

**storage\_backend**

**storage\_file\_id**

```
class indico.modules.events.abstracts.models.persons.AbstractPersonLink(*args,
                                                                    **kwargs)
    Bases: indico.modules.events.models.persons.PersonLinkBase
```

Association between EventPerson and Abstract.

**abstract\_id**

**author\_type**

**display\_order**

**id**

**is\_speaker**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**object\_relationship\_name** = 'abstract'

**person**

**person\_id**

**person\_link\_backref\_name** = 'abstract\_links'

**person\_link\_unique\_columns** = ('abstract\_id',)

```
class indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion(**kwargs):
    Bases: indico.core.db.sqlalchemy.review_questions.ReviewQuestionMixin,
           sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**description**

**event**

**event\_backref\_name** = 'abstract\_review\_questions'

**event\_id**

**field**

**field\_data**

**field\_type**

**id**

**is\_deleted**

**is\_required**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**position**

**title**

```
class indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating (**kwargs)
    Bases: indico.core.db.sqlalchemy.review_ratings.ReviewRatingMixin,
            sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

**question**

**question\_class**

alias of `indico.modules.events.abstracts.models.review_questions.  
AbstractReviewQuestion`

**question\_id**

**review**

**review\_class**

alias of `indico.modules.events.abstracts.models.reviews.AbstractReview`

**review\_id**

**value**

```
class indico.modules.events.abstracts.models.reviews.AbstractAction
```

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**accept = 1**

**change\_tracks = 3**

**mark\_as\_duplicate = 4**

**merge = 5**

```
reject = 2
```

```
class indico.modules.events.abstracts.models.reviews.AbstractCommentVisibility
    Bases: indico.util.enum.RichIntEnum
```

Most to least restrictive visibility for abstract comments.

```
contributors = 4
```

```
conveners = 2
```

```
judges = 1
```

```
reviewers = 3
```

```
users = 5
```

```
class indico.modules.events.abstracts.models.reviews.AbstractReview(**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalReviewMixin, indico.core.db.sqlalchemy.descriptions.RenderModeMixin, sqlalchemy.orm.decl_api.Model
```

An abstract review, emitted by a reviewer.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
abstract
```

```
abstract_id
```

```
can_edit (user, check_state=False)
```

```
can_view (user)
```

```
comment
```

```
created_dt
```

```
default_render_mode = 2
```

```
group_attr = 'track'
```

```
id
```

```
locator
```

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```
marshmallow_aliases = {'_comment': 'comment'}
modified_dt
possible_render_modes = {<RenderMode.markdown: 2>}
proposed_action
proposed_contribution_type
proposed_contribution_type_id
proposed_related_abstract
proposed_related_abstract_id
proposed_tracks
render_mode = 2
revision_attr = 'abstract'
score
track
track_id
user
user_id
visibility
```

## Operations

```
indico.modules.events.abstracts.operations.add_abstract_files(abstract, files,
                                                             log_action=True)
indico.modules.events.abstracts.operations.close_cfa(event)
indico.modules.events.abstracts.operations.create_abstract(event, abstract_data, custom_fields_data=None,
                                                           send_notifications=False,
                                                           submitter=None,
                                                           is_invited=False)
indico.modules.events.abstracts.operations.create_abstract_comment(abstract,
                                                                    comment_data)
indico.modules.events.abstracts.operations.create_abstract_review(abstract, track, user, review_data,
                                                                    questions_data)
indico.modules.events.abstracts.operations.delete_abstract(abstract,
                                                           delete_contrib=False)
indico.modules.events.abstracts.operations.delete_abstract_comment(comment)
```



```

indico.modules.events.abstracts.operations.delete_abstract_files (abstract,
                                                                    files)
indico.modules.events.abstracts.operations.judge_abstract (abstract, abstract_data,
                                                            judgment, judge, contrib_session=None,
                                                            merge_persons=False,
                                                            send_notifications=False)
indico.modules.events.abstracts.operations.open_cfa (event)
indico.modules.events.abstracts.operations.reset_abstract_state (abstract)
indico.modules.events.abstracts.operations.schedule_cfa (event, start_dt, end_dt,
                                                           modification_end_dt)
indico.modules.events.abstracts.operations.update_abstract (abstract, abstract_data, custom_fields_data=None)
indico.modules.events.abstracts.operations.update_abstract_comment (comment, comment_data)
indico.modules.events.abstracts.operations.update_abstract_review (review, review_data, questions_data)
indico.modules.events.abstracts.operations.update_reviewed_for_tracks (abstract, tracks)
indico.modules.events.abstracts.operations.withdraw_abstract (abstract)

```

## Utilities

```

indico.modules.events.abstracts.util.build_default_email_template (event,
                                                                    tpl_type)
    Build a default e-mail template based on a notification type provided by the user.
indico.modules.events.abstracts.util.can_create_invited_abstracts (event)
indico.modules.events.abstracts.util.clear_boa_cache (event)
    Delete the cached book of abstract.
indico.modules.events.abstracts.util.create_boa (event)
    Create the book of abstracts if necessary.
    Returns The path to the PDF file
indico.modules.events.abstracts.util.create_boa_tex (event)
    Create the book of abstracts as a LaTeX archive.
    Returns A BytesIO containing the zip file.
indico.modules.events.abstracts.util.create_mock_abstract (event)
    Create a mock abstract that can be used in previews.
    Brace for geek references.
indico.modules.events.abstracts.util.filter_field_values (fields, can_manage,
                                                           owns_abstract)

```

```
indico.modules.events.abstracts.util.generate_spreadsheet_from_abstracts(abstracts,  
                                                                           static_item_ids,  
                                                                           dynamic_items)
```

Generate a spreadsheet data from a given abstract list.

#### Parameters

- **abstracts** – The list of abstracts to include in the file
- **static\_item\_ids** – The abstract properties to be used as columns
- **dynamic\_items** – Contribution fields as extra columns

```
indico.modules.events.abstracts.util.get_events_with_abstract_persons(user,  
                                                                       dt=None)
```

Return a dict of event ids and the abstract submission related roles the user has in that event.

#### Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

```
indico.modules.events.abstracts.util.get_events_with_abstract_reviewer_convener(user,  
                                                                                dt=None)
```

Return a dict of event ids and the abstract reviewing related roles the user has in that event.

#### Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

```
indico.modules.events.abstracts.util.get_track_reviewer_abstract_counts(event,  
                                                                        user)
```

Get the numbers of abstracts per track for a specific user.

Note that this does not take into account if the user is a reviewer for a track; it just checks whether the user has reviewed an abstract in a track or not.

**Returns** A dict mapping tracks to dicts containing the counts.

```
indico.modules.events.abstracts.util.get_user_abstracts(event, user)
```

Get the list of abstracts where the user is a reviewer/convener.

```
indico.modules.events.abstracts.util.get_user_tracks(event, user)
```

Get the list of tracks where the user is a reviewer/convener.

```
indico.modules.events.abstracts.util.get_visible_reviewed_for_tracks(abstract,  
                                                                      user)
```

```
indico.modules.events.abstracts.util.has_user_tracks(event, user)
```

```
indico.modules.events.abstracts.util.make_abstract_form(event, user, notification_option=False,  
                                                         management=False,  
                                                         invited=False)
```

Extend the abstract WtForm to add the extra fields.

Each extra field will use a field named `custom_ID`.

#### Parameters

- **event** – The *Event* for which to create the abstract form.
- **user** – The user who is going to use the form.

- **notification\_option** – Whether to add a field to the form to disable triggering notifications for the abstract submission.
- **management** – Whether the form is used in the management area
- **invited** – Whether the form is used to create an invited abstract

**Returns** An *AbstractForm* subclass.

## Placeholders

```
class indico.modules.events.abstracts.placeholders.EventTitlePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The title of the event'
```

```
name = 'event_title'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.EventURLPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The URL of the event'
```

```
name = 'event_url'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The ID of the abstract'
```

```
name = 'abstract_id'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The title of the abstract'
```

```
name = 'abstract_title'
```

```
classmethod render(abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
advanced = True
```

```
description = 1'The direct URL of the abstract'
```

```
name = 'abstract_url'
```

```
classmethod render(abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractInvitationURLPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The link to submit an invited abstract'
```

```
name = 'invitation_url'
```

```
classmethod render(abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractTrackPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The name of the destination track'
```

```
name = 'abstract_track'
```

```
classmethod render(abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.AbstractSessionPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The name of the destination session'
```

```
name = 'abstract_session'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.PrimaryAuthorsPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The names of the primary authors (separated by commas)'
```

```
name = 'primary_authors'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.CoAuthorsPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The names of the co-authors (separated by commas)'
```

```
name = 'co_authors'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The full name of the submitter, no title'
```

```
name = 'submitter_name'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.SubmitterFirstNamePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
advanced = True
```

```
description = l'The first name of the submitter'
```

```
name = 'submitter_first_name'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.SubmitterLastNamePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
advanced = True
```

```
description = l'The last name of the submitter'
```

```
name = 'submitter_last_name'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = l'The title of the submitter (Dr, Prof., etc...)'
```

```
name = 'submitter_title'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.TargetAbstractIDPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = l'The ID of the target abstract (merge or duplicate)'
```

```
name = 'target_abstract_id'
```

```
classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.TargetAbstractTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'The title of the target abstract (merge or duplicate)'
```

```
    name = 'target_abstract_title'
```

```
    classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    advanced = True
```

```
    description = 1"The full name of the target abstract's submitter, no title (merge or duplicate)"
```

```
    name = 'target_submitter_name'
```

```
    classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.TargetSubmitterFirstNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    advanced = True
```

```
    description = 1"The first name of the target abstract's submitter (merge or duplicate)"
```

```
    name = 'target_submitter_first_name'
```

```
    classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.TargetSubmitterLastNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    advanced = True
```

```
    description = 1"The last name of the target abstract's submitter (merge or duplicate)"
```

```
    name = 'target_submitter_last_name'
```

```
    classmethod render (abstract)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.JudgmentCommentPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = 1'Comments written by event organizer (upon final decision)'
    name = 'judgment_comment'

    classmethod render (abstract)
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.ContributionTypePlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = 1'The contribution type that is associated to the abstract'
    name = 'contribution_type'

    classmethod render (abstract)
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder
    Bases: indico.util.placeholders.Placeholder

    advanced = True

    description = 1'Contribution URL'
    name = 'contribution_url'

    classmethod render (abstract)
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

## Settings

```
class indico.modules.events.abstracts.settings.AllowEditingType
    Bases: indico.util.enum.RichEnum

    An enumeration.
```



```

    submitter = 'submitter'
    submitter_all = 'submitter_all'
    submitter_authors = 'submitter_authors'
    submitter_primary = 'submitter_primary'
class indico.modules.events.abstracts.settings.BOACorrespondingAuthorType
    Bases: indico.util.enum.RichEnum

    An enumeration.

    none = 'none'

    speakers = 'speakers'
    submitter = 'submitter'
class indico.modules.events.abstracts.settings.BOALinkFormat
    Bases: indico.util.enum.RichEnum

    LaTeX book of abstracts link format setting.

    value is a 2-tuple of strings: first is the hyperref option to use second sets additional tex commands

    colorlinks = ('[colorlinks]', '')
    frame = ('', '')
    unstyled = ('[hidelinks]', '')
class indico.modules.events.abstracts.settings.BOASortField
    Bases: indico.util.enum.RichEnum

    An enumeration.

    abstract_title = 'title'
    board_number = 'board_number'
    id = 'id'
    schedule = 'schedule'
    schedule_board_number = 'schedule_board_number'
    session_board_number = 'session_board_number'
    session_schedule_board = 'session_schedule_board'
    session_title = 'session_title'
    speaker = 'speaker'
class indico.modules.events.abstracts.settings.SubmissionRightsType
    Bases: indico.util.enum.RichEnum

    An enumeration.

    all = 'all'
    speakers = 'speakers'

```

### 7.1.3 Agreement

---

**Todo:** Docstrings (module, models, utilities)

---

#### Models

```
class indico.modules.events.agreements.models.agreements.Agreement (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    Agreements between a person and Indico.

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    accept (from_ip, reason=None, on_behalf=False)

    accepted

    attachment
        Attachment

    attachment_filename
        Filename and extension of the attachment

    belongs_to (person)

    static create_from_data (event, type_, person)

    data
        Definition-specific data of the agreement

    definition

    event
        The Event this agreement is associated with

    event_id
        ID of the event

    id
        Entry ID

    identifier
        Unique identifier within the event and type

    is_orphan ()

    locator

    pending

    person_email
        Email of the person agreeing

    person_name
        Full name of the person agreeing
```

**reason**  
Explanation as to why the agreement was accepted/rejected

**reject** (*from\_ip*, *reason=None*, *on\_behalf=False*)

**rejected**

**render** (*form*, *\*\*kwargs*)

**reset** ()

**signed\_dt**  
The date and time the agreement was signed

**signed\_from\_ip**  
The IP from which the agreement was signed

**signed\_on\_behalf**

**state**  
A *AgreementState*

**timestamp**  
The date and time the agreement was created

**type**  
Type of agreement

**user**  
The user this agreement is linked to

**user\_id**  
ID of a linked user

**uuid**  
Entry universally unique ID

```
class indico.modules.events.agreements.models.agreements.AgreementState
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    accepted = 1
    accepted_on_behalf = 3
        agreement accepted on behalf of the person
    pending = 0
    rejected = 2
    rejected_on_behalf = 4
        agreement rejected on behalf of the person
```

## Utilities

```
indico.modules.events.agreements.util.get_agreement_definitions()
indico.modules.events.agreements.util.send_new_agreements(event, name, people, email_body,
                                                         cc_addresses,
                                                         from_address)
```

Create and send agreements for a list of people on a given event.

### Parameters

- **event** – The *Event* associated with the agreement
- **name** – The agreement type matching a `AgreementDefinition` name
- **people** – The list of people for whom agreements will be created
- **email\_body** – The body of the email
- **cc\_addresses** – Email addresses to send CCs to
- **from\_address** – Email address of the sender

### Placeholders

```
class indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder
```

```
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'Link to the agreement page'
```

```
    name = 'agreement_link'
```

```
    classmethod render (definition, agreement)
```

```
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

Parameters **kwargs** – arguments specific to the placeholder's context

```
    required = True
```

```
class indico.modules.events.agreements.placeholders.PersonNamePlaceholder
```

```
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'Name of the person'
```

```
    name = 'person_name'
```

```
    classmethod render (definition, agreement)
```

```
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

Parameters **kwargs** – arguments specific to the placeholder's context

## 7.1.4 Contribution

---

**Todo:** Docstrings (module, models, operations, utilities)

---

## Models

```
class indico.modules.events.contributions.models.contributions.Contribution(**kwargs)
    Bases: indico.core.db.sqlalchemy.searchable.SearchableTitleMixin, indico.
    core.db.sqlalchemy.descriptions.SearchableDescriptionMixin, indico.
    core.db.sqlalchemy.protection.ProtectionManagersMixin, indico.core.db.
    sqlalchemy.locations.LocationMixin, indico.core.db.sqlalchemy.attachments.
    AttachedItemsMixin, indico.core.db.sqlalchemy.notes.AttachedNotesMixin,
    indico.modules.events.models.persons.PersonLinkDataMixin, indico.modules.
    events.models.persons.AuthorsSpeakersMixin, indico.modules.events.
    contributions.models.contributions.CustomFieldsMixin, sqlalchemy.orm.
    decl_api.Model
```

```
ATTACHMENT_FOLDER_ID_COLUMN = 'contribution_id'
```

```
PRELOAD_EVENT_ATTACHED_ITEMS = True
```

```
PRELOAD_EVENT_NOTES = True
```

```
abstract
```

```
abstract_id
```

```
access_key = None
```

```
acl_entries
```

```
classmethod allocate_friendly_ids(event, n)
```

Allocate n Contribution friendly\_ids.

This is needed so that we can allocate all IDs in one go. Not doing so could result in DB deadlocks. All operations that create more than one contribution should use this method.

### Parameters

- **event** – the Event in question
- **n** – the number of ids to pre-allocate

```
allow_relationship_preloading = True
```

```
allowed_types_for_editable
```

```
board_number
```

```
can_manage(user, permission=None, allow_admin=True, check_parent=True, ex-
    plicit_permission=False)
```

Check if the user can manage the object.

### Parameters

- **user** – The *User* to check. May be None if the user is not logged in.
- **allow\_admin** – If admin users should always have access
- **check\_parent** – If the parent object should be checked. In this case the permission is ignored; only full management access is inherited to children.
- **explicit\_permission** – If the specified permission should be checked explicitly instead of short-circuiting the check for Indico admins or managers. When this option is set to **True**, the values of *allow\_admin* and *check\_parent* are ignored. This also applies if *permission* is **None** in which case this argument being set to **True** is equivalent to *allow\_admin* and *check\_parent* being set to **False**.

**Param** permission: The management permission that is needed for the check to succeed. If not specified, full management privs are required. May be set to the string 'ANY' to check if the user has any management privileges. If the user has *full\_access* privileges, he's assumed to have all possible permissions.

**can\_submit\_proceedings** (*user*)

Whether the user can submit editables/papers.

**code**

**default\_render\_mode** = 2

**disallowed\_protection\_modes** = frozenset()

**duration**

**duration\_display**

The displayed duration of the contribution.

This is the duration of the poster session if applicable, otherwise the duration of the contribution itself.

**duration\_poster**

**enabled\_editables**

Return all submitted editables with enabled types.

**end\_dt**

**end\_dt\_display**

The displayed end time of the contribution.

This is the end time of the poster session if applicable, otherwise the end time of the contribution itself.

**end\_dt\_poster**

**event**

**event\_id**

**field\_values**

Data stored in abstract/contribution fields

**friendly\_id**

The human-friendly ID for the contribution

**get\_editable** (*editable\_type*)

Get the editable of the given type.

**get\_non\_inheriting\_objects** ()

Get a set of child objects that do not inherit protection.

**has\_published\_editables**

**id**

**inherit\_location**

**inheriting\_have\_acl** = True

**is\_deleted**

**is\_paper\_reviewer** (*user*)

**is\_scheduled**

**is\_user\_associated** (*user*, *check\_abstract=False*)

**keywords**

**location\_backref\_name** = 'contributions'

**location\_parent**

The parent object to consult if the location is inherited.

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**log** (\*args, \*\*kwargs)

Log with prefilled metadata for the contribution.

**note**

**own\_address**

**own\_no\_access\_contact** = None

**own\_room**

**own\_room\_id**

**own\_room\_name**

**own\_venue**

**own\_venue\_id**

**own\_venue\_name**

**paper**

**paper\_content\_reviewers**

Paper content reviewers

**paper\_judges**

Paper reviewing judges

**paper\_layout\_reviewers**

Paper layout reviewers

**pending\_paper\_files**

Paper files not submitted for reviewing

**person\_links**

Persons associated with this contribution

**possible\_render\_modes** = {<RenderMode.html: 1>, <RenderMode.markdown: 2>}

**classmethod** `preload_acl_entries` (*event*)

**protection\_mode**

**protection\_parent**

The parent object to consult for `ProtectionMode.inheriting`.

**references**

External references associated with this contribution

**render\_mode**

**session**

**session\_block**

**session\_block\_id**

**session\_id**

**slug**

**start\_dt**

**start\_dt\_display**

The displayed start time of the contribution.

This is the start time of the poster session if applicable, otherwise the start time of the contribution itself.

**start\_dt\_poster**

**subcontribution\_count**

**subcontributions**

**submitters**

**title**

**track**

**track\_id**

**type**

**type\_id**

**verbose\_title**

**class** `indico.modules.events.contributions.models.contributions.CustomFieldsMixin`

Bases: `object`

Methods to process custom field data.

**get\_field\_value** (*field\_id*, *raw=False*)

**set\_custom\_field** (*field\_id*, *field\_value*)

**class** `indico.modules.events.contributions.models.fields.ContributionField` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**description**



**event****event\_id****field****field\_data****field\_type****filter\_choices****id****is\_active****is\_public****is\_required****is\_user\_editable****legacy\_id****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**mgmt\_field****position****title****visibility**

```
class indico.modules.events.contributions.models.fields.ContributionFieldValue(**kwargs)
    Bases: indico.modules.events.contributions.models.fields.ContributionFieldValueBase
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**contribution\_field**

```
contribution_field_backref_name = 'contribution_values'
contribution_field_id
contribution_id
data
```

```
class indico.modules.events.contributions.models.fields.ContributionFieldValueBase (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
contribution_field = <RelationshipProperty at 0x7fbaff9f6c40; no key>
contribution_field_backref_name = None
    The name of the backref on the ContributionField
contribution_field_id
data = Column(None, JSONB(astext_type=Text()), table=None, nullable=False)
friendly_data
```

```
class indico.modules.events.contributions.models.fields.ContributionFieldVisibility
    Bases: indico.util.enum.RichIntEnum
```

An enumeration.

```
managers_and_submitters = 2
managers_only = 3
public = 1
```

```
class indico.modules.events.contributions.models.persons.AuthorType
    Bases: int, indico.util.enum.IndicoEnum
```

An enumeration.

```
get_highest = <bound method AuthorType.get_highest of <enum 'AuthorType'>>
none = 0
primary = 1
secondary = 2
```

```
class indico.modules.events.contributions.models.persons.ContributionPersonLink (*args,
                                                                                   **kwargs)
    Bases: indico.modules.events.models.persons.PersonLinkBase
```

Association between EventPerson and Contribution.

```
author_type
contribution_id
display_order
id
is_author
is_speaker
```

**is\_submitter**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**object\_relationship\_name** = 'contribution'

**person**

**person\_id**

**person\_link\_backref\_name** = 'contribution\_links'

**person\_link\_unique\_columns** = ('contribution\_id',)

```
class indico.modules.events.contributions.models.persons.SubContributionPersonLink(*args,
                                                                                   **kwargs)
```

Bases: *indico.modules.events.models.persons.PersonLinkBase*

Association between EventPerson and SubContribution.

**author\_type** = 0

**display\_order**

**id**

**is\_speaker** = True

**object\_relationship\_name** = 'subcontribution'

**person**

**person\_id**

**person\_link\_backref\_name** = 'subcontribution\_links'

**person\_link\_unique\_columns** = ('subcontribution\_id',)

**subcontribution\_id**

```
class indico.modules.events.contributions.models.principals.ContributionPrincipal(**kwargs)
Bases: indico.core.db.sqlalchemy.principals.PrincipalPermissionsMixin,
sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_category_roles = True
allow_emails = True
allow_event_roles = True
allow_registration_forms = True
category_role
category_role_id
contribution_id
    The ID of the associated contribution
disallowed_protection_modes = frozenset()
email
event_role
event_role_id
full_access
id
    The ID of the acl entry
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
permissions
principal_backref_name = 'in_contribution_acls'
principal_for = 'Contribution'
read_access
registration_form
registration_form_id
type
unique_columns = ('contribution_id',)
user
user_id
```

```
class indico.modules.events.contributions.models.references.ContributionReference(**kwargs)
    Bases: indico.modules.events.models.references.ReferenceModelBase
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

contribution_id
id
reference_backref_name = 'contribution_references'
reference_type
reference_type_id
value

```

```

class indico.modules.events.contributions.models.references.SubContributionReference (**kwargs)
    Bases: indico.modules.events.models.references.ReferenceModelBase

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

id
reference_backref_name = 'subcontribution_references'
reference_type
reference_type_id
subcontribution_id
value

```

```

class indico.modules.events.contributions.models.subcontributions.SubContribution (**kwargs)
    Bases: indico.core.db.sqlalchemy.searchable.SearchableTitleMixin, indico.
    core.db.sqlalchemy.descriptions.SearchableDescriptionMixin, indico.core.db.
    sqlalchemy.attachments.AttachedItemsMixin, indico.core.db.sqlalchemy.notes.
    AttachedNotesMixin, sqlalchemy.orm.decl_api.Model

```

```

ATTACHMENT_FOLDER_ID_COLUMN = 'subcontribution_id'

```

```

PRELOAD_EVENT_ATTACHED_ITEMS = True

```

```

PRELOAD_EVENT_NOTES = True

```

```

can_access (user, **kwargs)

```

```

can_manage (user, permission=None, **kwargs)

```

```

code

```

```

contribution_id

```

```

default_render_mode = 2

```

```

duration

```

```

event

```

```

friendly_id

```

The human-friendly ID for the sub-contribution

```

get_access_list ()

```

```

get_manager_list (recursive=False, include_groups=True)

```

**id**

**is\_deleted**

**is\_protected**

**is\_user\_associated** (*user*)

**location\_parent**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**note**

**person\_links**

Persons associated with this contribution

**position**

**possible\_render\_modes** = {<RenderMode.html: 1>, <RenderMode.markdown: 2>}

**references**

External references associated with this contribution

**render\_mode**

**room\_name**

**session**

Convenience property so all event entities have it.

**slug**

**speakers**

**timetable\_entry**

Convenience property so all event entities have it.

**title**

**venue\_name**

**class** `indico.modules.events.contributions.models.types.ContributionType` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**description**

**event**

**event\_id**

**id**

**is\_private**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**name**

## Operations

```
indico.modules.events.contributions.operations.create_contribution(event,
                                                                    con-
                                                                    trib_data,
                                                                    cus-
                                                                    tom_fields_data=None,
                                                                    ses-
                                                                    sion_block=None,
                                                                    ex-
                                                                    tend_parent=False)

indico.modules.events.contributions.operations.create_contribution_from_abstract(abstract,
                                                                                   con-
                                                                                   trib_session=

indico.modules.events.contributions.operations.create_subcontribution(contrib,
                                                                        data)

indico.modules.events.contributions.operations.delete_contribution(contrib)

indico.modules.events.contributions.operations.delete_subcontribution(subcontrib)
```

```
indico.modules.events.contributions.operations.update_contribution (contrib,
                                                                    con-
                                                                    trib_data,
                                                                    cus-
                                                                    tom_fields_data=None)
```

Update a contribution.

#### Parameters

- **contrib** – The *Contribution* to update
- **contrib\_data** – A dict containing the data to update
- **custom\_fields\_data** – A dict containing the data for custom fields.

**Returns** A dictionary containing information related to the update. *unscheduled* will be true if the modification resulted in the contribution being unscheduled. In this case *undo\_unschedule* contains the necessary data to re-schedule it (undoing the session change causing it to be unscheduled)

```
indico.modules.events.contributions.operations.update_subcontribution (subcontrib,
                                                                    data)
```

## Utilities

```
indico.modules.events.contributions.util.contribution_type_row (contrib_type)
```

```
indico.modules.events.contributions.util.generate_spreadsheet_from_contributions (contributions)
```

Return a tuple consisting of spreadsheet columns and respective contribution values.

```
indico.modules.events.contributions.util.get_boa_export_formats ()
```

```
indico.modules.events.contributions.util.get_contributions_for_person (event,
                                                                    per-
                                                                    son,
                                                                    only_speakers=False)
```

Get all contributions for an event person.

If *only\_speakers* is true, then only contributions where the person is a speaker are returned

```
indico.modules.events.contributions.util.get_contributions_with_user_as_submitter (event,
                                                                    user)
```

Get a list of contributions in which the *user* has submission rights.

```
indico.modules.events.contributions.util.get_events_with_linked_contributions (user,
                                                                    dt=None)
```

Return a dict with keys representing *event\_id* and the values containing data about the user rights for contributions within the event.

#### Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

```
indico.modules.events.contributions.util.has_contributions_with_user_as_submitter (event,
                                                                    user)
```

```
indico.modules.events.contributions.util.import_contributions_from_csv (event,
                                                                    f)
```

Import timetable contributions from a CSV file into an event.



```
indico.modules.events.contributions.util.make_contribution_form(event)
```

Extend the contribution WtForm to add the extra fields.

Each extra field will use a field named `custom_ID`.

**Parameters** **event** – The *Event* for which to create the contribution form.

**Returns** A *ContributionForm* subclass.

```
indico.modules.events.contributions.util.render_archive(event, contribs, sort_by,
                                                         cls)
```

```
indico.modules.events.contributions.util.render_pdf(event, contribs, sort_by, cls)
```

```
indico.modules.events.contributions.util.serialize_contribution_for_ical(contrib)
```

```
indico.modules.events.contributions.util.serialize_contribution_person_link(person_link,
                                                                               is_submitter=None)
```

Serialize ContributionPersonLink to JSON-like object.

```
indico.modules.events.contributions.util.sort_contribs(contribs, sort_by)
```

## 7.1.5 Feature

---

**Todo:** Docstrings (module, utilities)

---

### Utilities

```
indico.modules.events.features.util.format_feature_names(names)
```

```
indico.modules.events.features.util.get_disallowed_features(event)
```

Get a set containing the names of features which are not available for an event.

```
indico.modules.events.features.util.get_enabled_features(event,
                                                           only_explicit=False)
```

Return a set of enabled feature names for an event.

```
indico.modules.events.features.util.get_feature_definition(name)
```

Get a feature definition.

```
indico.modules.events.features.util.get_feature_definitions()
```

Get a dict containing all feature definitions.

```
indico.modules.events.features.util.is_feature_enabled(event, name)
```

Check if a feature is enabled for an event.

#### Parameters

- **event** – The event (or event ID) to check.
- **name** – The name of the feature.

```
indico.modules.events.features.util.require_feature(event, name)
```

Raise a `NotFound` error if a feature is not enabled.

#### Parameters

- **event** – The event (or event ID) to check.
- **name** – The name of the feature.

```
indico.modules.events.features.util.set_feature_enabled(event, name, state)
```

Enable/disable a feature for an event.

#### Parameters

- **event** – The event.
- **name** – The name of the feature.
- **state** – If the feature is enabled or not.

**Returns** Boolean indicating if the state of the feature changed.

## 7.1.6 Layout

---

**Todo:** Docstrings (module, models, utilities)

---

### Models

```
class indico.modules.events.layout.models.images.ImageField(**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

#### **content\_type**

The MIME type of the file.

#### **created\_dt**

The date/time when the file was uploaded.

#### **event**

#### **event\_id**

The event the image belongs to

#### **extension**

The extension of the file.

#### **filename**

The name of the file.

#### **id**

The ID of the file

#### **locator**

#### **md5**

An MD5 hash of the file.

Automatically assigned when *save()* is called.

**size**  
 The size of the file (in bytes).  
 Automatically assigned when *save()* is called.

**storage\_backend**

**storage\_file\_id**

**version\_of = None**

**class** `indico.modules.events.layout.models.menu.EventPage(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**event**  
 The Event which contains the page

**event\_id**  
 The ID of the event which contains the page

**html**  
 The rendered HTML of the page

**id**  
 The ID of the page

**is\_default**

**locator**

**class** `indico.modules.events.layout.models.menu.MenuEntry(**kwargs)`

Bases: `indico.modules.events.layout.models.menu.MenuEntryMixin`, `sqlalchemy.orm.decl_api.Model`

**children**  
 The children menu entries and parent backref

**event**  
 The Event containing the menu entry

**event\_id**  
 The ID of the event which contains the menu

**static get\_for\_event** (*event*)

**id**  
 The ID of the menu entry

**insert** (*parent*, *position*)

**is\_enabled**  
 Whether the entry is visible in the event's menu

**is\_root**

**link\_url**  
 The target URL of a custom link

**move** (*to*)

**name**  
The name of the menu entry (to uniquely identify a default entry for a given event)

**new\_tab**  
Whether the menu entry should be opened in a new tab or window

**page**  
The page of the menu entry

**page\_id**  
The page ID if the entry is a page

**parent\_id**  
The ID of the parent menu entry (NULL if root menu entry)

**plugin**  
The name of the plugin from which the entry comes from (NULL if the entry does not come from a plugin)

**position**  
The relative position of the entry in the menu

**registered\_only**  
Whether the menu entry should be viewable only by registered users

**title**  
The title of the menu entry (to be displayed to the user)

**type**  
The type of the menu entry

```
class indico.modules.events.layout.models.menu.MenuEntryMixin (**kwargs)
```

Bases: `object`

**default\_data**

**event\_ref**

**is\_internal\_link**

**is\_link**

**is\_orphaned**

**is\_page**

**is\_plugin\_link**

**is\_separator**

**is\_user\_link**

**is\_visible**

**localized\_title**

**locator**

**url**

```
class indico.modules.events.layout.models.menu.MenuEntryType
```

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**internal\_link** = 2

**page** = 5

```

plugin_link = 4
separator = 1
user_link = 3

class indico.modules.events.layout.models.menu.TransientMenuEntry(event,
                                                                    is_enabled,
                                                                    name,
                                                                    position,
                                                                    children)

Bases: indico.modules.events.layout.models.menu.MenuEntryMixin

id

```

## Utilities

```

class indico.modules.events.layout.util.MenuEntryData(title, name, endpoint=
point=None, position=-1, is_enabled=True, visible=None, parent=None, static_site=False,
url_kwargs=None, hide_if_restricted=True)

Bases: object

```

Container to transmit menu entry-related data via signals.

The data contained is transmitted via the *sidemenu* signal and used to build the side menu of an event.

### Parameters

- **title** – str – The title of the menu, displayed to the user. The title should be translated using the normal gettext function, i.e. `_( ' . . . ' )`, or the plugin’s bound gettext function.
- **name** – str – Name used to refer to the entry internally. This is never shown to the user. The name must be unique, names from plugins are automatically prefixed with the plugin name and a colon and therefore have to be unique only within the plugin. To mark the entry as active, its name must be specified in the *menu\_entry\_name* class attribute of the WP class. For plugins, the plugin name must be specified via the *menu\_entry\_plugin* attribute as well.
- **endpoint** – str – The endpoint the entry will point to.
- **position** – int – The desired position of the menu entry. the position is indicative only, relative to the other entries and not the exact position. Entries with the same position will be sorted alphanumerically on their name. A position of *-1* will append the entry at the end of the menu.
- **is\_enabled** – bool – Whether the entry should be enabled by default (Default: *True*).
- **visible** – function – Determines if the entry should be visible. This is a simple function which takes only the *event* as parameter and returns a boolean to indicate if the entry is visible or not. It is called whenever the menu is displayed, so the current state of the event/user can be taken into account.
- **parent** – str – The name of the parent entry (None for root entries).
- **static\_site** – bool or str – If True, this menu item should be shown in the menu of a static site. When set to a string, the string will be used instead of a mangled version of the endpoint’s URL.

- **url\_kwargs** – dict – Additional data passed to `url_for` when building the url the menu item points to.

**name**

**plugin** = None

**visible** (*event*)

`indico.modules.events.layout.util.build_menu_entry_name(name, plugin=None)`

Build the proper name for a menu entry.

Given a menu entry's name and optionally a plugin, returns the correct name of the menu entry.

#### Parameters

- **name** – str – The name of the menu entry.
- **plugin** – IndicoPlugin or str – The plugin (or the name of the plugin) which created the entry.

`indico.modules.events.layout.util.get_css_file_data(event)`

`indico.modules.events.layout.util.get_css_url(event, force_theme=None, for_preview=False)`

Build the URL of a CSS resource.

#### Parameters

- **event** – The *Event* to get the CSS url for
- **force\_theme** – The ID of the theme to override the custom CSS resource only if it exists
- **for\_preview** – Whether the URL is used in the CSS preview page

**Returns** The URL to the CSS resource

`indico.modules.events.layout.util.get_logo_data(event)`

`indico.modules.events.layout.util.get_menu_entries_from_signal()`

`indico.modules.events.layout.util.get_menu_entry_by_name(name, event)`

`indico.modules.events.layout.util.get_plugin_conference_themes()`

`indico.modules.events.layout.util.is_menu_entry_enabled(entry_name, event)`

Check whether the MenuEntry is enabled.

`indico.modules.events.layout.util.menu_entries_for_event(event)`

## 7.1.7 Log

---

**Todo:** Docstrings (module, models, utilities)

---

### Models

**class** `indico.modules.events.logs.models.entries.EventLogEntry(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

Log entries for events.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**data**

Type-specific data

**event**

The Event this log entry is associated with

**event\_id**

The ID of the event

**id**

The ID of the log entry

**kind**

The general kind of operation that was performed

**logged\_date**

**logged\_dt**

The date/time when the reminder was created

**meta**

Non-displayable data

**module**

The module the operation was related to (does not need to match something in `indico.modules` and should be human-friendly but not translated).

**realm**

The general area of the event the entry comes from

**render ()**

Render the log entry to be displayed.

If the renderer is not available anymore, e.g. because of a disabled plugin, `None` is returned.

**renderer**

**summary**

A short one-line description of the logged action. Should not be translated!

**type**

The type of the log entry. This needs to match the name of a log renderer.

**user**

The user associated with the log entry

**user\_id**

The ID of the user associated with the entry

**class** `indico.modules.events.logs.models.entries.EventLogKind`

Bases: `int`, `indico.util.enum.IndicoEnum`

An enumeration.

**change** = 3

**negative** = 4

**other** = 1

**positive** = 2

```
class indico.modules.events.logs.models.entries.EventLogRealm
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    emails = 5
    event = 1
    management = 2
    participants = 3
    reviewing = 4
```

## Utilities

```
indico.modules.events.logs.util.get_log_renderers()
```

```
indico.modules.events.logs.util.make_diff_log(changes, fields)
    Create a value for log data containing change information.
```

### Parameters

- **changes** – a dict mapping attributes to (old, new) tuples
- **fields** – a dict mapping attributes to field metadata. for simple cases this may be a string with the human-friendly title, for more advanced fields it should be a dict containing `title`, a `type` string and a `convert` callback which will be invoked with a tuple containing the old and new value

```
indico.modules.events.logs.util.render_changes(a, b, type_)
    Render the comparison of a and b as HTML.
```

### Parameters

- **a** – old value
- **b** – new value
- **type** – the type determining how the values should be compared

```
indico.modules.events.logs.util.serialize_log_entry(entry)
```

```
class indico.modules.events.logs.renderers.EmailRenderer
    Bases: indico.modules.events.logs.renderers.EventLogRendererBase

    name = 'email'

    template_name = 'events/logs/entry_email.html'
```

```
class indico.modules.events.logs.renderers.EventLogRendererBase
    Bases: object
```

Base class for event log renderers.

```
classmethod get_data(entry)
```

Return the entry data in a format suitable for the template.

This method may be overridden if the entry's data needs to be preprocessed before being passed to the template.

It MUST NOT modify *entry.data* directly.

```
name = None
```

unique name of the log renderer (matches EventLogEntry.type)



```

plugin = None
    plugin containing this renderer - assigned automatically

classmethod render_entry (entry)
    Render the log entry row.

    Parameters entry – A EventLogEntry

template_kwargs = {}
    extra kwargs passed to render_template

template_name = None
    template used to render the log entry

class indico.modules.events.logs.renderers.SimpleRenderer
    Bases: indico.modules.events.logs.renderers.EventLogRendererBase

    classmethod get_data (entry)
        Return the entry data in a format suitable for the template.

        This method may be overridden if the entry's data needs to be preprocessed before being passed to the
        template.

        It MUST NOT modify entry.data directly.

    name = 'simple'

    template_kwargs = {'compare': <function render_changes>}

    template_name = 'events/logs/entry_simple.html'

```

### 7.1.8 Event Management

```

class indico.modules.events.management.controllers.RHManageEventBase
    Bases: indico.modules.events.controllers.base.RHEventBase, indico.modules.events.management.controllers.base.ManageEventMixin

    Base class for event management RHs.

class indico.modules.events.management.views.WPEventManager (rh, event_, active_menu_item=None,
                                                                **kwargs)
    Bases: indico.web.views.WPJinjaMixin, indico.web.views.WPDecorated

    Base class for event management pages.

    When using this class the template will always have event available; it is not necessary to pass it as a kwarg
    when calling the render_template classmethod.

    When using the class directly, pass the menu item as a posarg:

```

```

return WPEventManager.render_template('foobar.html', self.event, 'foobar',
                                       foo='bar')

```

When subclassing you can set *sidemenu\_option* on the class, allowing you to omit it. This is recommended if you have many pages using the same menu item or if you already need to subclass for some other reason (e.g. to set a *template\_prefix* or include additional JS/CSS bundles):

```

return WPSomething.render_template('foobar.html', self.event,
                                    foo='bar')

```

## 7.1.9 Note

---

**Todo:** Docstrings (module, models, utilities)

---

### Models

```
class indico.modules.events.notes.models.notes.EventNote (**kwargs)
    Bases: indico.core.db.sqlalchemy.links.LinkMixin, sqlalchemy.orm.decl_api.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allowed_link_types = frozenset({<LinkType.event: 2>, <LinkType.contribution: 3>, <Li
```

```
category = None
```

```
category_id = None
```

```
contribution
```

```
contribution_id
```

```
create_revision (render_mode, source, user)
```

Create a new revision if needed and marks it as undeleted if it was.

Any change to the render mode or the source causes a new revision to be created. The user is not taken into account since a user “modifying” a note without changing things is not really a change.

```
current_revision
```

The currently active revision of the note

```
current_revision_id
```

The ID of the current revision

```
delete (user)
```

Mark the note as deleted and adds a new empty revision.

```
event
```

```
event_id
```

```
events_backref_name = 'all_notes'
```

```
classmethod get_for_linked_object (linked_object, preload_event=True)
```

Get the note for the given object.

This only returns a note that hasn't been deleted.

#### Parameters

- **linked\_object** – An event, session, contribution or subcontribution.
- **preload\_event** – If all notes for the same event should be pre-loaded and cached in the app context.

**classmethod** `get_or_create` (*linked\_object*)

Get the note for the given object or creates a new one.

If there is an existing note for the object, it will be returned even. Otherwise a new note is created.

**html**

The rendered HTML of the note

**classmethod** `html_matches` (*search\_string*, *exact=False*)

Check whether the html content matches a search string.

To be used in a SQLAlchemy *filter* call.

#### Parameters

- **search\_string** – A string to search for
- **exact** – Whether to search for the exact string

**id**

The ID of the note

**is\_deleted**

If the note has been deleted

**link\_backref\_name** = 'note'

**link\_type**

**linked\_event**

**linked\_event\_id**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**revisions**

The list of all revisions for the note

**session**

**session\_block** = None

**session\_block\_id** = None

**session\_id**

**subcontribution**

**subcontribution\_id**

**unique\_links = True**

**class** indico.modules.events.notes.models.notes.**EventNoteRevision** (*\*\*kwargs*)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

**created\_dt**

The date/time when the revision was created

**html**

The rendered HTML of the note

**id**

The ID of the revision

**note**

**note\_id**

The ID of the associated note

**render\_mode**

How the note is rendered

**source**

The raw source of the note as provided by the user

**user**

The user who created the revision

**user\_id**

The user who created the revision

## Utilities

indico.modules.events.notes.util.**build\_note\_api\_data** (*note*)

indico.modules.events.notes.util.**build\_note\_legacy\_api\_data** (*note*)

indico.modules.events.notes.util.**can\_edit\_note** (*obj, user*)

Check if a user can edit the object’s note.

indico.modules.events.notes.util.**get\_scheduled\_notes** (*event*)

Get all notes of scheduled items inside an event.

### 7.1.10 Paper

---

**Todo:** Docstrings (module, models, operations, utilities, settings)

---

The `papers` module handles the Indico’s Paper Peer Reviewing workflow. The “inputs” of this module are the conference papers, which will be uploaded by the corresponding authors/submitters.

## Models

**class** indico.modules.events.papers.models.call\_for\_papers.**CallForPapers** (*event*)

Bases: `object`

Proxy class to facilitate access to the call for papers settings.

**announcement**

**assignees**

**can\_access\_judging\_area** (*user*)

**can\_access\_reviewing\_area** (*user*)

**close** ()

**content\_review\_questions**

**content\_reviewer\_deadline**

**content\_reviewer\_deadline\_enforced**

**content\_reviewers**

**content\_reviewing\_enabled**

**end\_dt**

**get\_questions\_for\_review\_type** (*review\_type*)

**get\_reviewing\_state** (*reviewing\_type*)

**has\_ended**

**has\_started**

**is\_judge** (*user*)

**is\_manager** (*user*)

**is\_open**

**is\_reviewer** (*user*, *role=None*)

**is\_staff** (*user*)

**judge\_deadline**

**judges**

**layout\_review\_questions**

**layout\_reviewer\_deadline**

**layout\_reviewer\_deadline\_enforced**

**layout\_reviewers**

**layout\_reviewing\_enabled**

**managers**

**open** ()

**rating\_range**

**schedule** (*start\_dt*, *end\_dt*)

**set\_reviewing\_state** (*reviewing\_type, enable*)

**start\_dt**

**user\_competences**

```
class indico.modules.events.papers.models.comments.PaperReviewComment (**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalCommentMixin, indico.
            core.db.sqlalchemy.review_comments.ReviewCommentMixin, sqlalchemy.orm.
            decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**can\_edit** (*user*)

**can\_view** (*user*)

**created\_dt**

**id**

**is\_deleted**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**modified\_by**

**modified\_by\_id**

**modified\_dt**

**paper\_revision**

**render\_mode** = 2

**revision\_id**

**user**

**user\_backref\_name** = 'review\_comments'

**user\_id**

```
user_modified_backref_name = 'modified_review_comments'
```

```
visibility
```

```
class indico.modules.events.papers.models.competences.PaperCompetence (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
competences
```

```
event
```

```
event_id
```

```
id
```

```
classmethod merge_users (target, source)
```

```
user
```

```
user_id
```

```
class indico.modules.events.papers.models.files.PaperFile (*args, **kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model
```

```
add_file_date_column = False
```

```
content_type
```

The MIME type of the file.

```
created_dt = None
```

```
extension
```

The extension of the file.

```
filename
```

The name of the file.

```
id
```

```
locator
```

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**md5**

An MD5 hash of the file.

Automatically assigned when `save()` is called.

**paper****paper\_revision****revision\_id****size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.

**storage\_backend****storage\_file\_id**

**class** `indico.modules.events.papers.models.papers.Paper` (*contribution*)

Bases: `indico.modules.events.models.reviews.ProposalMixin`

Proxy class to facilitate access to all paper-related properties.

**accepted\_revision**

**call\_for\_proposals\_attr** = `'cfp'`

**can\_comment** (*user*, *check\_state=False*)

**can\_judge** (*user*, *check\_state=False*)

**can\_manage** (*user*)

**can\_review** (*user*, *check\_state=False*)

**can\_submit** (*user*)

**event****files**

**get\_last\_revision** ()

**get\_revisions** ()

**is\_in\_final\_state****judgment\_comment****last\_revision****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:



```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```

proposal_type = 'paper'
proxied_attr = 'contribution'
reset_state()
revision_count
revisions
revisions_enabled = True
state
title
verbose_title

```

```

class indico.modules.events.papers.models.review_questions.PaperReviewQuestion(**kwargs)
    Bases: indico.core.db.sqlalchemy.review_questions.ReviewQuestionMixin,
           sqlalchemy.orm.decl_api.Model

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

description
event
event_backref_name = 'paper_review_questions'
event_id
field
field_data
field_type
id
is_deleted
is_required
locator

```

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**position**

**title**

**type**

```
class indico.modules.events.papers.models.review_ratings.PaperReviewRating(**kwargs)
    Bases: indico.core.db.sqlalchemy.review_ratings.ReviewRatingMixin,
           sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

**question**

**question\_class**

alias of `indico.modules.events.papers.models.review_questions.PaperReviewQuestion`

**question\_id**

**review**

**review\_class**

alias of `indico.modules.events.papers.models.reviews.PaperReview`

**review\_id**

**value**

```
class indico.modules.events.papers.models.reviews.PaperAction
```

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**accept = 1**

**reject = 2**

**to\_be\_corrected = 3**

```
class indico.modules.events.papers.models.reviews.PaperCommentVisibility
```

Bases: `indico.util.enum.RichIntEnum`

Most to least restrictive visibility for paper comments.

```
contributors = 3
judges = 1
reviewers = 2
users = 4
```

```
class indico.modules.events.papers.models.reviews.PaperJudgmentProxy(paper)
    Bases: object
```

A timeline item for the non final judgments.

```
created_dt
timeline_item_type = 'judgment'
```

```
class indico.modules.events.papers.models.reviews.PaperReview(**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalReviewMixin, indico.core.db.sqlalchemy.descriptions.RenderModeMixin, sqlalchemy.orm.decl_api.Model
```

A paper review, emitted by a layout or content reviewer.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
TIMELINE_TYPE = 'review'
can_edit (user, check_state=False)
can_view (user)
comment
created_dt
default_render_mode = 2
group_attr = 'type'
group_proxy_cls
    alias of PaperTypeProxy
id
locator
```

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
```

(continues on next page)

(continued from previous page)

```
def locator(self):  
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**modified\_dt**

**possible\_render\_modes** = {<RenderMode.markdown: 2>}

**proposed\_action**

**render\_mode** = 2

**revision**

**revision\_attr** = 'revision'

**revision\_id**

**score**

**type**

**user**

**user\_id**

**visibility**

**class** `indico.modules.events.papers.models.reviews.PaperReviewType`

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**content** = 2

**layout** = 1

**class** `indico.modules.events.papers.models.reviews.PaperTypeProxy(group)`

Bases: `indico.modules.events.models.reviews.ProposalGroupProxy`

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property  
def locator(self):  
    return {...}  
  
@locator.other  
def locator(self):  
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```

class indico.modules.events.papers.models.revisions.PaperRevision(*args,
                                                                    **kwargs)
    Bases: indico.modules.events.models.reviews.ProposalRevisionMixin, indico.
            core.db.sqlalchemy.descriptions.RenderModeMixin, sqlalchemy.orm.decl_api.
            Model

    default_render_mode = 2

    get_reviewed_for_groups(user, include_reviewed=False)

    get_reviews(group=None, user=None)

    get_spotlight_file()

    get_timeline(user=None)

    has_user_reviewed(user, review_type=None)

    id

    is_last_revision

    judge

    judge_id

    judgment_comment

    judgment_dt

    locator
        Define a smart locator property.

        This behaves pretty much like a normal read-only property and the decorated function should return a dict
        containing the necessary data to build a URL for the object.

        This decorator should usually be applied to a method named locator as this name is required for
        get_locator to find it automatically when just passing the object.

        If you need more than one locator, you can define it like this:

        @locator_property
        def locator(self):
            return {...}

        @locator.other
        def locator(self):
            return {...}

    number

    paper

    possible_render_modes = {<RenderMode.markdown: 2>}

    proposal_attr = 'paper'

    render_mode = 2

    spotlight_file

    state

    submitted_dt

```

**submitter**

**submitter\_id**

**timeline**

**class** indico.modules.events.papers.models.revisions.**PaperRevisionState**

Bases: indico.util.enum.RichIntEnum

An enumeration.

**accepted = 2**

**rejected = 3**

**submitted = 1**

**to\_be\_corrected = 4**

**class** indico.modules.events.papers.models.templates.**PaperTemplate** (\*\*kwargs)

Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**add\_file\_date\_column = False**

**content\_type**

The MIME type of the file.

**created\_dt = None**

**description**

**event**

**event\_id**

**extension**

The extension of the file.

**filename**

The name of the file.

**id**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}
```

(continues on next page)

(continued from previous page)

```
@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**md5**

An MD5 hash of the file.

Automatically assigned when `save()` is called.

**name****size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.

**storage\_backend****storage\_file\_id**

## Operations

`indico.modules.events.papers.operations.close_cfp(event)`

`indico.modules.events.papers.operations.create_comment(paper, text, visibility, user)`

`indico.modules.events.papers.operations.create_competences(event, user, competences)`

`indico.modules.events.papers.operations.create_paper_revision(paper, submitter, files)`

`indico.modules.events.papers.operations.create_paper_template(event, data)`

`indico.modules.events.papers.operations.create_review(paper, review_type, user, review_data, questions_data)`

`indico.modules.events.papers.operations.delete_comment(comment)`

`indico.modules.events.papers.operations.delete_paper_template(template)`

`indico.modules.events.papers.operations.judge_paper(paper, judgment, comment, judge)`

`indico.modules.events.papers.operations.open_cfp(event)`

`indico.modules.events.papers.operations.reset_paper_state(paper)`

`indico.modules.events.papers.operations.schedule_cfp(event, start_dt, end_dt)`

`indico.modules.events.papers.operations.set_deadline(event, role, deadline, enforce=True)`

`indico.modules.events.papers.operations.set_reviewing_state(event, reviewing_type, enable)`

`indico.modules.events.papers.operations.update_comment(comment, text=None, visibility=None)`

`indico.modules.events.papers.operations.update_competences(user_competences, competences)`

```
indico.modules.events.papers.operations.update_paper_template(template, data)
indico.modules.events.papers.operations.update_review(review, review_data, ques-
                                                    tions_data)
indico.modules.events.papers.operations.update_reviewing_roles(event, users,
                                                                contributions,
                                                                role, assign)
indico.modules.events.papers.operations.update_team_members(event, managers,
                                                            judges, content_reviewers=None,
                                                            lay-
                                                            out_reviewers=None)
```

## Utilities

```
indico.modules.events.papers.util.get_contributions_with_paper_submitted_by_user(event,
                                                                                      user)
```

```
indico.modules.events.papers.util.get_events_with_paper_roles(user, dt=None)
    Get the IDs and PR roles of events where the user has any kind of paper reviewing privileges.
```

### Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

**Returns** A dict mapping event IDs to a set of roles

```
indico.modules.events.papers.util.get_user_contributions_to_review(event,
                                                                    user)
```

Get the list of contributions where user has paper to review.

```
indico.modules.events.papers.util.get_user_reviewed_contributions(event,
                                                                    user)
```

Get the list of contributions where user already reviewed paper.

```
indico.modules.events.papers.util.get_user_submittable_contributions(event,
                                                                    user)
```

```
indico.modules.events.papers.util.has_contributions_with_user_paper_submission_rights(event,
                                                                                          user)
```

```
indico.modules.events.papers.util.is_type_reviewing_possible(cfp, review_type)
```

## Settings

```
class indico.modules.events.settings.EventACLProxy(proxy)
    Bases: indico.core.settings.proxy.ACLProxyBase
```

Proxy class for event-specific ACL settings.

```
add_principal(event, name, principal)
    Add a principal to an ACL.
```

### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*



**contains\_user** (*event, name, user*)

Check if a user is in an ACL.

To pass this check, the user can either be in the ACL itself or in a group in the ACL.

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **user** – A *User*

**get** (*event, name*)

Retrieves an ACL setting

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name

**merge\_users** (*target, source*)

Replace all ACL user entries for *source* with *target*.

**remove\_principal** (*event, name, principal*)

Remove a principal from an ACL.

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

**set** (*event, name, acl*)

Replace an ACL with a new one.

#### Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **acl** – A set containing principals (users/groups)

```
class indico.modules.events.settings.EventSettingProperty (proxy, name, default=<object object>, attr=None)
```

Bases: `indico.core.settings.proxy.SettingProperty`

```
attr = 'event'
```

```
class indico.modules.events.settings.EventSettingsProxy (module, defaults=None, strict=True, acls=None, converters=None)
```

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access event-specific settings for a certain module.

**acl\_proxy\_class**

alias of *EventACLProxy*

**delete** (*event, \*names*)

Delete settings.

#### Parameters

- **event** – Event (or its ID)
- **names** – One or more names of settings to delete

**delete\_all** (*event*)  
Delete all settings.

**Parameters** **event** – Event (or its ID)

**get** (*event, name, default=<object object>*)  
Retrieve the value of a single setting.

**Parameters**

- **event** – Event (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

**Returns** The settings's value or the default value

**get\_all** (*event, no\_defaults=False*)  
Retrieve all settings.

**Parameters**

- **event** – Event (or its ID)
- **no\_defaults** – Only return existing settings and ignore defaults.

**Returns** Dict containing the settings

**query**  
Return a query object filtering by the proxy's module.

**set** (*event, name, value*)  
Set a single setting.

**Parameters**

- **event** – Event (or its ID)
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

**set\_multi** (*event, items*)  
Set multiple settings at once.

**Parameters**

- **event** – Event (or its ID)
- **items** – Dict containing the new settings

**class** `indico.modules.events.settings.ThemeSettingsProxy`  
Bases: `object`

**defaults**

**get\_themes\_for** (*event\_type*)

**settings**

**themes**

`indico.modules.events.settings.event_or_id(f)`

### 7.1.11 Payment

---

**Todo:** Docstrings (module, models, plugins)

---

#### Models

**exception** `indico.modules.events.payment.models.transactions.DoublePaymentTransaction`  
 Bases: `Exception`

**exception** `indico.modules.events.payment.models.transactions.IgnoredTransactionAction`  
 Bases: `Exception`

**exception** `indico.modules.events.payment.models.transactions.InvalidManualTransactionAction`  
 Bases: `Exception`

**exception** `indico.modules.events.payment.models.transactions.InvalidTransactionAction`  
 Bases: `Exception`

**exception** `indico.modules.events.payment.models.transactions.InvalidTransactionStatus`  
 Bases: `Exception`

**class** `indico.modules.events.payment.models.transactions.PaymentTransaction` (*\*\*kwargs*)  
 Bases: `sqlalchemy.orm.decl_api.Model`

Payment transactions.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**amount**  
 the base amount the user needs to pay (without payment-specific fees)

**classmethod** `create_next` (*registration, amount, currency, action, provider=None, data=None*)

**currency**  
 the currency of the payment (ISO string, e.g. EUR or USD)

**data**  
 plugin-specific data of the payment

**id**  
 Entry ID

**is\_manual**

**plugin**

**provider**  
 the provider of the payment (e.g. manual, PayPal etc.)

**registration**  
 The associated registration

**registration\_id**  
 ID of the associated registration

```
render_details()
    Render the transaction details.

status
    a TransactionStatus

timestamp
    the date and time the transaction was recorded

class indico.modules.events.payment.models.transactions.TransactionAction
    Bases: int, indico.util.enum.IndicoEnum

    An enumeration.

    cancel = 2
    complete = 1
    pending = 3
    reject = 4

class indico.modules.events.payment.models.transactions.TransactionStatus
    Bases: int, indico.util.enum.IndicoEnum

    An enumeration.

    cancelled = 2
        payment cancelled manually
    failed = 3
        payment attempt failed
    pending = 4
        payment on hold pending approval of merchant
    rejected = 5
        payment rejected after being pending
    successful = 1
        payment attempt succeeded

class indico.modules.events.payment.models.transactions.TransactionStatusTransition
    Bases: object

    initial_statuses = [<TransactionStatus.cancelled: 2>, <TransactionStatus.failed: 3>,
    classmethod next (transaction, action, provider=None)
```

## Utilities

```
indico.modules.events.payment.util.get_active_payment_plugins (event)
    Return a dict containing the active payment plugins of an event.

indico.modules.events.payment.util.get_payment_plugins ()
    Return a dict containing the available payment plugins.

indico.modules.events.payment.util.register_transaction (registration, amount,
                                                         currency, action,
                                                         provider=None,
                                                         data=None)

    Create a new transaction for a certain transaction action.
```

### Parameters

- **registration** – the *Registration* associated to the transaction
- **amount** – the (strictly positive) amount of the transaction
- **currency** – the currency used for the transaction
- **action** – the *TransactionAction* of the transaction
- **provider** – the payment method name of the transaction, or ‘\_manual’ if no payment method has been used
- **data** – arbitrary JSON-serializable data specific to the transaction’s provider

## Plugins

**class** `indico.modules.events.payment.plugins.PaymentPluginMixin`

Bases: `object`

**adjust\_payment\_form\_data** (*data*)

Update the payment form data if necessary.

This method can be overridden to update e.g. the amount based on choices the user makes in the payment form or to provide additional data to the form. To do so, *data* must be modified.

**Parameters** *data* – a dict containing event, registration, amount, currency, settings and event\_settings

**can\_be\_modified** (*user, event*)

Check if the user is allowed to enable/disable/modify the payment method.

**Parameters**

- **user** – the *User* representing the user
- **event** – the *Event*

**category** = 'Payment'

**default\_settings**

**event\_settings\_form**

alias of `PaymentEventSettingsFormBase`

**get\_event\_management\_url** (*event, \*\*kwargs*)

**get\_invalid\_regforms** (*event*)

Return registration forms with incompatible currencies.

**get\_method\_name** (*event*)

Return the (customized) name of the payment method.

**init** ()

**logo\_url**

**render\_payment\_form** (*registration*)

Return the payment form shown to the user.

**Parameters** *registration* – a *Registration* object

**render\_transaction\_details** (*transaction*)

Render the transaction details in event management.

Override this (or inherit from the template) to show more useful data such as transaction IDs

**Parameters** *transaction* – the *PaymentTransaction*

**settings\_form**  
alias of `PaymentPluginSettingsFormBase`

**supports\_currency** (*currency*)

**valid\_currencies** = `None`  
Set containing all valid currencies. Set to *None* to allow all.

## 7.1.12 Person

---

**Todo:** Docstrings (module, operations)

---

### Operations

`indico.modules.events.persons.operations.update_person` (*person*, *data*)

### Placeholders

**class** `indico.modules.events.persons.placeholders.ContributionsPlaceholder`  
Bases: `indico.util.placeholders.ParametrizedPlaceholder`

**classmethod** `iter_param_info` (*person*, *event*, *\*\*kwargs*)  
Yield information for known params.

Each item yielded must be a (*value*, *description*) tuple.

**Parameters** *kwargs* – arguments specific to the placeholder’s context

**name** = `'contributions'`

**param\_required** = `False`

**param\_restricted** = `True`

**classmethod** `render` (*param*, *person*, *event*, *\*\*kwargs*)  
Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using *\*\*kwargs*.

**Parameters** *kwargs* – arguments specific to the placeholder’s context

**class** `indico.modules.events.persons.placeholders.EmailPlaceholder`  
Bases: `indico.util.placeholders.Placeholder`

**description** = `l'Email of the person'`

**name** = `'email'`

**classmethod** `render` (*person*, *event*, *\*\*kwargs*)  
Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using *\*\*kwargs*.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.persons.placeholders.EventLinkPlaceholder
```

Bases: `indico.util.placeholders.Placeholder`

**description** = `1'Link to the event'`

**name** = `'event_link'`

**classmethod** **render** (*person, event, \*\*kwargs*)

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.persons.placeholders.EventTitlePlaceholder
```

Bases: `indico.util.placeholders.Placeholder`

**description** = `1'The title of the event'`

**name** = `'event_title'`

**classmethod** **render** (*person, event, \*\*kwargs*)

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.persons.placeholders.FirstNamePlaceholder
```

Bases: `indico.util.placeholders.Placeholder`

**description** = `1'First name of the person'`

**name** = `'first_name'`

**classmethod** **render** (*person, event, \*\*kwargs*)

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.persons.placeholders.LastNamePlaceholder
```

Bases: `indico.util.placeholders.Placeholder`

**description** = `1'Last name of the person'`

**name** = `'last_name'`

**classmethod** **render** (*person, event, \*\*kwargs*)

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.events.persons.placeholders.RegisterLinkPlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'The link for the registration page'
```

```
    name = 'register_link'
```

```
    classmethod render (person, event, **kwargs)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** **kwargs** – arguments specific to the placeholder’s context

## 7.1.13 Registration

---

**Todo:** Docstrings (module, models, utilities, statistics)

---

### Models

```
class indico.modules.events.registration.models.registrations.Registration (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

Somebody’s registration for an event through a registration form.

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

**avatar\_url**

Return the url of the user’s avatar.

**base\_price**

The base registration fee (that is not specific to form items)

**billable\_data**

**can\_be\_modified**

**can\_be\_withdrawn**

**checked\_in**

Whether the person has checked in. Setting this also sets or clears *checked\_in\_dt*.

**checked\_in\_dt**

The date/time when the person has checked in

**currency**

Registration price currency

**data**

The registration this data is associated with



**data\_by\_field**

**display\_full\_name**

Return the full name using the user's preferred name format.

**email**

The email of the registrant

**event**

The Event containing this registration

**event\_id**

The ID of the event

**first\_name**

The first name of the registrant

**friendly\_id**

The human-friendly ID for the object

**full\_name**

Return the user's name in 'Firstname Lastname' notation.

**classmethod get\_all\_for\_event** (*event*)

Retrieve all registrations in all registration forms of an event.

**get\_full\_name** (*last\_name\_first=True, last\_name\_upper=False, abbrev\_first\_name=False*)

Return the user's in the specified notation.

If not format options are specified, the name is returned in the 'Lastname, Firstname' notation.

Note: Do not use positional arguments when calling this method. Always use keyword arguments!

#### Parameters

- **last\_name\_first** – if “lastname, firstname” instead of “firstname lastname” should be used
- **last\_name\_upper** – if the last name should be all-uppercase
- **abbrev\_first\_name** – if the first name should be abbreviated to use only the first character

**get\_personal\_data** ()

**has\_conflict** ()

Check if there are other valid registrations for the same user.

This is intended for cases where this registration is currently invalid (rejected or withdrawn) to determine whether it would be acceptable to restore it.

**has\_files**

**id**

The ID of the object

**is\_active**

**is\_cancelled**

**is\_deleted**

If the registration has been deleted

**is\_paid**

Return whether the registration has been paid for.

**is\_publishable**

**is\_ticket\_blocked**

Check whether the ticket is blocked by a plugin.

**last\_name**

The last name of the registrant

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**log (\*args, \*\*kwargs)**

Log with prefilled metadata for the registration.

**classmethod merge\_users (target, source)**

**order\_by\_name** = (<sqlalchemy.sql.functions.Function at 0x7fbafe364d60; lower>, <sqlalchemy.sql.functions.Function at 0x7fbafe364d60; lower>)

**payment\_dt**

The date/time when the registration has been paid for.

**price**

The total price of the registration.

This includes the base price, the field-specific price, and the custom price adjustment for the registrant.

**Return type** Decimal

**price\_adjustment**

The price modifier applied to the final calculated price

**registration\_form\_id**

The ID of the registration form

**rejection\_reason**

If given a reason for rejection

**render\_base\_price ()**

**render\_price ()**

**render\_price\_adjustment ()**

**sections\_with\_answered\_fields**

**state**

The state a registration is in

**submitted\_dt**

The date/time when the registration was recorded

**summary\_data**

Export registration data nested in sections and fields.

**sync\_state** (*\_skip\_moderation=True*)

Sync the state of the registration.

**ticket\_uuid**

The unique token used in tickets

**transaction**

The latest payment transaction associated with this registration

**transaction\_id**

The ID of the latest payment transaction associated with this registration

**update\_state** (*approved=None, paid=None, rejected=None, withdrawn=None, \_skip\_moderation=False*)

Update the state of the registration for a given action.

The accepted kwargs are the possible actions. `True` means that the action occurred and `False` that it was reverted.

**user****user\_id**

The ID of the user who registered

**uuid**

The unguessable ID for the object

```
class indico.modules.events.registration.models.registrations.RegistrationData (**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model
```

Data entry within a registration for a field in a registration form.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**add\_file\_date\_column = False****content\_type**

The MIME type of the file.

**created\_dt = None****data**

The submitted data for the field

**extension**

The extension of the file.

**field\_data**

The associated field data object

**field\_data\_id**

The ID of the field data

**file**

**file\_required = False**

**filename**

The name of the file.

**friendly\_data**

**get\_friendly\_data** (*\*\*kwargs*)

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**md5**

An MD5 hash of the file.

Automatically assigned when `save()` is called.

**price**

**registration\_id**

The ID of the registration

**render\_price** ()

**search\_data**

**size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.

**storage\_backend**

**storage\_file\_id**

**summary\_data**

**user\_data**

```
class indico.modules.events.registration.models.registrations.RegistrationState
Bases: indico.util.enum.RichIntEnum
```

An enumeration.

```
complete = 1
pending = 2
rejected = 3
unpaid = 5
withdrawn = 4
```

```
class indico.modules.events.registration.models.form_fields.RegistrationFormField(**kwargs)
    Bases: indico.modules.events.registration.models.items.RegistrationFormItem
```

A registration form field.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
calculate_price (registration_data)
```

```
children
```

```
current_data
```

```
current_data_id
```

```
data
```

```
data_versions
```

```
description
```

```
field_impl
```

Gets the implementation of the field.

**Returns** An instance of a *RegistrationFormFieldBase* subclass

```
get_friendly_data (registration_data, **kwargs)
```

```
html_field_name
```

```
id
```

```
input_type
```

```
is_deleted
```

```
is_enabled
```

```
is_manager_only
```

```
is_required
```

```
locator
```

```
parent_id
```

```
personal_data_type
```

```
position
```

```
registration_form_id
```

```
title
```

**type**

**versioned\_data**

**view\_data**

Return object with data that Angular can understand.

**class** `indico.modules.events.registration.models.form_fields.RegistrationFormFieldData` (\*\*kwargs)

Bases: `sqlalchemy.orm.decl_api.Model`

Description of a registration form field.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**field\_id**

The ID of the registration form field

**id**

The ID of the object

**versioned\_data**

Data describing the field

**class** `indico.modules.events.registration.models.form_fields.RegistrationFormPersonalDataField`

Bases: `indico.modules.events.registration.models.form_fields.RegistrationFormField`

*RegistrationFormField*

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**children**

**current\_data**

**current\_data\_id**

**data**

**data\_versions**

**description**

**html\_field\_name**

**id**

**input\_type**

**is\_deleted**

**is\_enabled**

**is\_manager\_only**

**is\_required**

**parent\_id**

**personal\_data\_type**

**position**

**registration\_form\_id**

**title**

**type**

**view\_data**

Return object with data that Angular can understand.

**class** `indico.modules.events.registration.models.forms.ModificationMode`

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**allowed\_always** = 1

**allowed\_until\_approved** = 4

**allowed\_until\_payment** = 2

**not\_allowed** = 3

**class** `indico.modules.events.registration.models.forms.RegistrationForm(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

A registration form for an event.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**active\_fields**

**active\_registrations**

**attach\_ical**

If the completed registration email should include the event's iCalendar file.

**base\_price**

The base fee users have to pay when registering

**can\_submit** (*user*)

**contact\_info**

Contact information for registrants

**currency**

Currency for prices in the registration form

**disabled\_sections**

**end\_dt**

Datetime when the registration form is closed

**event**

The Event containing this registration form

**event\_id**

The ID of the event

**form\_items**

**get\_personal\_data\_field\_id** (*personal\_data\_type*)  
Return the field id corresponding to the personal data field with the given name.

**get\_registration** (*user=None, uuid=None, email=None*)  
Retrieve registrations for this registration form by user or uuid.

**has\_ended**

**has\_started**

**id**  
The ID of the object

**identifier**

**introduction**

**invitations**  
The registration invitations associated with this form

**is\_active**

**is\_deleted**  
Whether the registration has been marked as deleted

**is\_modification\_allowed** (*registration*)  
Check whether a registration may be modified.

**is\_modification\_open**

**is\_open**

**is\_participation**  
Whether it's the 'Participants' form of a meeting/lecture

**is\_scheduled**

**limit\_reached**

**locator**

**manager\_notification\_recipients**  
List of emails that should receive management notifications

**manager\_notifications\_enabled**  
Whether the manager notifications for this event are enabled

**message\_complete**  
Custom message to include in emails for complete registrations

**message\_pending**  
Custom message to include in emails for pending registrations

**message\_unpaid**  
Custom message to include in emails for unpaid registrations

**moderation\_enabled**  
Whether registrations must be approved by a manager

**modification\_end\_dt**  
Datetime when the modification period is over

**modification\_mode**  
Whether registration modifications are allowed

**name**



**notification\_sender\_address**  
Notifications sender address

**principal\_order = 2**

**principal\_type = 8**

**publish\_checkin\_enabled**  
Whether checked-in status should be displayed in the event pages and participant list

**publish\_registration\_count**  
Whether to display the number of registrations

**publish\_registrations\_enabled**  
Whether registrations should be displayed in the participant list

**registration\_limit**  
Maximum number of registrations allowed

**registrations**  
The registrations associated with this form

**render\_base\_price ()**

**require\_login**  
Whether users must be logged in to register

**require\_user**  
Whether registrations must be associated with an Indico account

**sections**

**start\_dt**  
Datetime when the registration form is open

**ticket\_on\_email**  
Whether to send tickets by e-mail

**ticket\_on\_event\_page**  
Whether to show a ticket download link on the event homepage

**ticket\_on\_summary\_page**  
Whether to show a ticket download link on the registration summary page

**ticket\_template**  
The template used to generate tickets

**ticket\_template\_id**  
The ID of the template used to generate tickets

**tickets\_enabled**  
Whether tickets are enabled for this form

**title**  
The title of the registration form

**class** `indico.modules.events.registration.models.invitations.InvitationState`  
Bases: `indico.util.enum.RichIntEnum`  
An enumeration.

**accepted = 1**

**declined = 2**

**pending = 0**

```
class indico.modules.events.registration.models.invitations.RegistrationInvitation (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

An invitation for someone to register.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**affiliation**

The affiliation of the invited person

**email**

The email of the invited person

**first\_name**

The first name of the invited person

**id**

The ID of the invitation

**last\_name**

The last name of the invited person

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**registration**

The associated registration

**registration\_form\_id**

The ID of the registration form

**registration\_id**

The ID of the registration (if accepted)

**skip\_moderation**

Whether registration moderation should be skipped

**state**

The state of the invitation

**uuid**

The UUID of the invitation

**class** `indico.modules.events.registration.models.items.PersonalDataType`

Bases: `int`, `indico.util.enum.IndicoEnum`

Description of the personal data items that exist on every registration form.

**FIELD\_DATA** = [`(<PersonalDataType.first_name: 2>`, `{'title': 'First Name', 'input_type`

`address = 6`

`affiliation = 4`

**column**

The Registration column in which the value is stored in addition to the regular registration data entry.

`country = 8`

`email = 1`

`first_name = 2`

`get_title()`

`is_required`

`last_name = 3`

`phone = 7`

`position = 9`

`title = 5`

**class** `indico.modules.events.registration.models.items.RegistrationFormItem(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

Generic registration form item.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**children**

**current\_data**

The latest value of the field

**current\_data\_id**

The ID of the latest data

**data**

unversioned field data

**data\_versions**

The list of all versions of the field data

**description**

Description of this field

**id**

The ID of the object

**input\_type**  
input type of this field

**is\_deleted**  
Whether field has been “deleted”

**is\_enabled**  
Whether the field is enabled

**is\_field**

**is\_manager\_only**  
if the section is only accessible to managers

**is\_required**  
determines if the field is mandatory

**is\_section**

**is\_visible**

**parent\_id**  
The ID of the parent form item

**personal\_data\_type**  
The type of a personal data field

**position**

**registration\_form\_id**  
The ID of the registration form

**title**  
The title of this field

**type**  
The type of the registration form item

**view\_data**  
Return object with data that Angular can understand.

```
class indico.modules.events.registration.models.items.RegistrationFormItemType
    Bases: int, indico.util.enum.IndicoEnum
```

An enumeration.

```
field = 2
field_pd = 5
section = 1
section_pd = 4
text = 3
```

```
class indico.modules.events.registration.models.items.RegistrationFormPersonalDataSection (
    Bases: indico.modules.events.registration.models.items.RegistrationFormSection
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

**children**  
**current\_data**  
**current\_data\_id**  
**data**  
**data\_versions**  
**description**  
**id**  
**input\_type**  
**is\_deleted**  
**is\_enabled**  
**is\_manager\_only**  
**is\_required**  
**parent\_id**  
**personal\_data\_type**  
**position**  
**registration\_form\_id**  
**title**  
**type**  
**view\_data**

Return object with data that Angular can understand.

**class** `indico.modules.events.registration.models.items.RegistrationFormSection` (*\*\*kwargs*)  
 Bases: `indico.modules.events.registration.models.items.RegistrationFormItem`

Registration form section that can contain fields and text.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**active\_fields**  
**children**  
**current\_data**  
**current\_data\_id**  
**data**  
**data\_versions**  
**description**  
**fields**  
**id**  
**input\_type**

`is_deleted`  
`is_enabled`  
`is_manager_only`  
`is_required`  
`locator`  
`own_data`  
`parent_id`  
`personal_data_type`  
`position`  
`registration_form_id`  
`title`  
`type`  
`view_data`

Return object with data that Angular can understand.

**class** `indico.modules.events.registration.models.items.RegistrationFormText` (*\*\*kwargs*)

Bases: `indico.modules.events.registration.models.items.RegistrationFormItem`

Text to be displayed in registration form sections.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

`children`  
`current_data`  
`current_data_id`  
`data`  
`data_versions`  
`description`  
`id`  
`input_type`  
`is_deleted`  
`is_enabled`  
`is_manager_only`  
`is_required`  
`locator`  
`parent_id`  
`personal_data_type`  
`position`

**registration\_form\_id**

**title**

**type**

**view\_data**

Return object with data that Angular can understand.

## Utilities

`indico.modules.events.registration.util.build_registration_api_data(registration)`

`indico.modules.events.registration.util.build_registrations_api_data(event)`

`indico.modules.events.registration.util.check_registration_email(regform, email, registration=None, management=False)`

Check whether an email address is suitable for registration.

### Parameters

- **regform** – The registration form
- **email** – The email address
- **registration** – The existing registration (in case of modification)
- **management** – If it's a manager adding a new registration

`indico.modules.events.registration.util.create_personal_data_fields(regform)`

Create the special section/fields for personal data.

`indico.modules.events.registration.util.create_registration(regform, data, invitation=None, management=False, notify_user=True, skip_moderation=None)`

`indico.modules.events.registration.util.generate_spreadsheet_from_registrations(registrations, regform_items, static_items)`

Generate a spreadsheet data from a given registration list.

### Parameters

- **registrations** – The list of registrations to include in the file
- **regform\_items** – The registration form items to be used as columns
- **static\_items** – Registration form information as extra columns

`indico.modules.events.registration.util.generate_ticket(registration)`

`indico.modules.events.registration.util.generate_ticket_qr_code(registration)`

Generate a Pillow *Image* with a QR Code encoding a check-in ticket.

**Parameters** **registration** – corresponding *Registration* object

```
indico.modules.events.registration.util.get_event_regforms(event, user,
                                                            with_registrations=False,
                                                            only_in_acl=False)
```

Get registration forms with information about user registrations.

**Parameters**

- **event** – the *Event* to get registration forms for
- **user** – A *User*
- **with\_registrations** – Whether to return the user’s registration instead of just whether they have one
- **only\_in\_acl** – Whether to include only registration forms that are in the event’s ACL

```
indico.modules.events.registration.util.get_event_regforms_registrations(event,
                                                                           user,
                                                                           in-
                                                                           clude_scheduled=True,
                                                                           only_in_acl=False)
```

Get regforms and the associated registrations for an event+user.

**Parameters**

- **event** – the *Event* to get registration forms for
- **user** – A *User*
- **include\_scheduled** – Whether to include scheduled but not open registration forms
- **only\_in\_acl** – Whether to include only registration forms that are in the event’s ACL

**Returns** A tuple, which includes: - All registration forms which are scheduled, open or registered.  
- A dict mapping all registration forms to the user’s registration if they have one.

```
indico.modules.events.registration.util.get_event_section_data(regform, management=False,
                                                                registra-
                                                                tion=None)
```

```
indico.modules.events.registration.util.get_events_registered(user, dt=None)
```

Get the IDs of events where the user is registered.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

**Returns** A set of event ids

```
indico.modules.events.registration.util.get_published_registrations(event)
```

Get a list of published registrations for an event.

**Parameters** **event** – the *Event* to get registrations for

**Returns** list of *Registration* objects

```
indico.modules.events.registration.util.get_registered_event_persons(event)
```

Get all registered EventPersons of an event.

```
indico.modules.events.registration.util.get_registrations_with_tickets(user,
                                                                           event)
```

```
indico.modules.events.registration.util.get_ticket_attachments(registration)
```



```
indico.modules.events.registration.util.get_title_uuid(regform, title)
```

Convert a string title to its UUID value.

If the title does not exist in the title PD field, it will be ignored and returned as None.

```
indico.modules.events.registration.util.import_registrations_from_csv(regform,
                                                                       fileobj,
                                                                       skip_moderation=True,
                                                                       no-
                                                                       tify_users=False)
```

Import event registrants from a CSV file into a form.

```
indico.modules.events.registration.util.make_registration_form(regform, management=False,
                                                                registra-
                                                                tion=None)
```

Create a WTForm based on registration form fields.

```
indico.modules.events.registration.util.modify_registration(registration, data,
                                                            management=False,
                                                            notify_user=True)
```

```
indico.modules.events.registration.util.serialize_registration_form(regform)
```

Serialize registration form to JSON-like object.

```
indico.modules.events.registration.util.update_regform_item_positions(regform)
```

Update positions when deleting/disabling an item in order to prevent gaps.

```
indico.modules.events.registration.util.url_rule_to_angular(endpoint)
```

Convert a flask-style rule to angular style.

## Placeholders

```
class indico.modules.events.registration.placeholders.registrations.EventLinkPlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
description = 1'Link to the event'
```

```
name = 'event_link'
```

```
classmethod render(regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.EventTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The title of the event'
```

```
name = 'event_title'
```

```
classmethod render(regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.FieldPlaceholder
    Bases: indico.util.placeholders.ParametrizedPlaceholder
```

```
advanced = True
```

```
description = None
```

```
classmethod iter_param_info (regform, registration)
```

Yield information for known params.

Each item yielded must be a (value, description) tuple.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
name = 'field'
```

```
param_required = True
```

```
param_restricted = True
```

```
classmethod render (param, regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.FirstNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
description = 1'First name of the person'
```

```
name = 'first_name'
```

```
classmethod render (regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.IDPlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
description = 1'The ID of the registration'
```

```
name = 'id'
```

```
classmethod render (regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.LastNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'Last name of the person'
```

```
    name = 'last_name'
```

```
    classmethod render (regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.LinkPlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'The link to the registration details'
```

```
    name = 'link'
```

```
    classmethod render (regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.registrations.RejectionReasonPlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'The reason why the registration was rejected'
```

```
    name = 'rejection_reason'
```

```
    classmethod render (regform, registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.invitations.FirstNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```
    description = 1'First name of the person'
```

```
    name = 'first_name'
```

```
    classmethod render (invitation)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.events.registration.placeholders.invitations.InvitationLinkPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = 1'Link to accept/decline the invitation'

    name = 'invitation_link'

    classmethod render (invitation)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.

        Parameters kwargs – arguments specific to the placeholder’s context

    required = True
```

```
class indico.modules.events.registration.placeholders.invitations.LastNamePlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = 1'Last name of the person'

    name = 'last_name'

    classmethod render (invitation)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.

        Parameters kwargs – arguments specific to the placeholder’s context
```

## Settings

```
class indico.modules.events.registration.settings.RegistrationSettingsProxy (module,
                                                                 de-
                                                                 faults=None,
                                                                 strict=True,
                                                                 acls=None,
                                                                 con-
                                                                 vert-
                                                                 ers=None)

    Bases: indico.modules.events.settings.EventSettingsProxy

    Store per-event registration settings.

    get_participant_list_columns (event, form=None)

    get_participant_list_form_ids (event)

    set_participant_list_columns (event, columns, form=None)

    set_participant_list_form_ids (event, form_ids)
```

## Statistics

**class** `indico.modules.events.registration.stats.AccommodationStats` (*field*)  
 Bases: `indico.modules.events.registration.stats.FieldStats`, `indico.modules.events.registration.stats.StatsBase`

**class** `indico.modules.events.registration.stats.Cell`  
 Bases: `indico.modules.events.registration.stats.Cell`

Hold data and type for a cell of a stats table.

The table below indicates the valid types and expected data.

type	data
<i>str</i>	<i>str</i> – string value
<i>progress</i>	( <i>int</i> , <i>str</i> ) – a tuple with the progress (a value between 0 and 1) and a label
<i>progress-stacked</i>	( <i>[int]</i> , <i>str</i> ) – a tuple with a list of progresses (values which must sum up to 1) and a label
<i>currency</i>	<i>float</i> – numeric value
<i>icon</i>	<i>str</i> – icon name from <code>_icons.scss</code>
<i>default</i>	<i>None</i> – renders a default cell with an <code>&amp;mdash;</code> ; (use <code>Cell(type='str')</code> for an empty cell)

### Parameters

- **type** – *str* – The type of data in the cell
- **data** – The data for the cell
- **colspan** – *int* – HTML colspan value for the cell
- **classes** – [*str*] – HTML classes to apply to the cell
- **qtip** – *str* – content for qtip

**class** `indico.modules.events.registration.stats.DataItem`  
 Bases: `indico.modules.events.registration.stats.DataItem`

Hold the aggregation of some data, intended for stats tables as a aggregation from which to generate cells.

### Parameters

- **regs** – *int* – number of registrant
- **attendance** – *int* – number of people attending
- **capacity** – *int* – maximum number of people allowed to attend (*0* if unlimited)
- **billable** – *bool* – whether the item is billable to the or not
- **cancelled** – *bool* – whether the item is cancelled or not
- **price** – *str* – the price of the item
- **fixed\_price** – *bool* – *True* if the price is per registrant, *False* if accompanying guests must pay as well.
- **paid** – *int* – number of registrants who paid
- **paid\_amount** – *float* – amount already paid by registrants
- **unpaid** – *int* – number of registrants who haven't paid
- **unpaid\_amount** – *float* – amount not already paid by registrants

```
class indico.modules.events.registration.stats.FieldStats (field, **kwargs)
    Bases: object

    Hold stats for a registration form field.

    get_table()
        Return a table containing the stats for each item.

        Returns dict – A table with a list of head cells (key: ‘head’) and a list of rows (key: ‘rows’)
            where each row is a list of cells.

    is_currency_shown
```

```
class indico.modules.events.registration.stats.OverviewStats (regform)
    Bases: indico.modules.events.registration.stats.StatsBase

    Generic stats for a registration form.
```

```
class indico.modules.events.registration.stats.StatsBase (title, subtitle, type, **kwargs)
    Bases: object

    Base class for registration form statistics.

    Parameters

    • title – str – the title for the stats box

    • subtitle – str – the subtitle for the stats box

    • type – str – the type used in Jinja to display the stats

    is_currency_shown
```

## 7.1.14 Reminder

---

**Todo:** Docstrings (module)

---

### Models

```
class indico.modules.events.reminders.models.reminders.EventReminder (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    Email reminders for events.

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any
    mapped columns or relationships.

    all_recipients
        Return all recipients of the notifications.

        This includes both explicit recipients and, if enabled, participants/speakers of the event.

    attach_ical
        If the notification should include the event’s iCalendar file.
```

**created\_dt**

The date/time when the reminder was created

**creator**

The user who created the reminder

**creator\_id**

The ID of the user who created the reminder

**event**

The Event this reminder is associated with

**event\_id**

The ID of the event

**event\_start\_delta**

How long before the event start the reminder should be sent This is needed to update the *scheduled\_dt* when changing the start time of the event.

**id**

The ID of the reminder

**include\_description**

If the notification should include the event's description.

**include\_summary**

If the notification should include a summary of the event's schedule.

**is\_overdue****is\_relative**

Return if the reminder is relative to the event time.

**is\_sent**

If the reminder has been sent

**locator****message**

Custom message to include in the email

**recipients**

The recipients of the notification

**reply\_to\_address**

The address to use as Reply-To in the notification email.

**scheduled\_dt**

The date/time when the reminder should be sent

**send()**

Send the reminder to its recipients.

**send\_to\_participants**

If the notification should also be sent to all event participants

**send\_to\_speakers**

If the notification should also be sent to all event speakers

## Utilities

`indico.modules.events.reminders.util.make_reminder_email(event, with_agenda, with_description, note)`

Return the template module for the reminder email.

### Parameters

- **event** – The event
- **with\_agenda** – If the event’s agenda should be included
- **note** – A custom message to include in the email

## 7.1.15 Request

---

**Todo:** Docstrings (module)

---

## Models

**class** `indico.modules.events.requests.models.requests.Request` (*\*\*kwargs*)  
Bases: `sqlalchemy.orm.decl_api.Model`

Event-related requests, e.g. for a webcast.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

### **can\_be\_modified**

Determine if the request can be modified or if a new one must be sent.

### **comment**

an optional comment for an accepted/rejected request

### **created\_by\_id**

ID of the user creating the request

### **created\_by\_user**

The user who created the request

### **created\_dt**

the date/time the request was created

### **data**

plugin-specific data of the request

### **definition**

### **event**

The Event this agreement is associated with

### **event\_id**

ID of the event

**classmethod** `find_latest_for_event` (*event, type\_=None*)

Return the latest requests for a given event.



**Parameters**

- **event** – the event to find the requests for
- **type** – the request type to retrieve, or *None* to get all

**Returns** a dict mapping request types to a *Request* or if *type\_* was specified, a single *Request* or *None*

**id**  
request ID

**locator**

**processed\_by\_id**  
ID of the user processing the request

**processed\_by\_user**  
The user who processed the request

**processed\_dt**  
the date/time the request was accepted/rejected

**state**  
the requests's date, a *RequestState* value

**type**  
the request type name

```
class indico.modules.events.requests.models.requests.RequestState
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    accepted = 1
    pending = 0
    rejected = 2
    withdrawn = 3
```

**Utilities**

```
indico.modules.events.requests.util.get_request_definitions()
    Return a dict of request definitions.
```

```
indico.modules.events.requests.util.is_request_manager(user)
    Check if the user manages any request types.
```

```
class indico.modules.events.requests.base.RequestDefinitionBase
    Bases: object
```

A service request which can be sent by event managers.

```
classmethod accept(req, data, user)
    Accept the request.
```

To ensure that additional data is saved, this method should call **:method:'manager\_save'**.

**Parameters**

- **req** – the *Request* of the request
- **data** – the form data from the management form

- **user** – the user processing the request

**classmethod can\_be\_managed** (*user*)

Check whether the user is allowed to manage this request type.

**Parameters** **user** – a *User*

**classmethod create\_form** (*event*, *existing\_request=None*)

Create the request form.

**Parameters**

- **event** – the event the request is for
- **existing\_request** – the *Request* if there's an existing request of this type

**Returns** an instance of an *IndicoForm* subclass

**classmethod create\_manager\_form** (*req*)

Create the request management form.

**Parameters** **req** – the *Request* of the request

**Returns** an instance of an *IndicoForm* subclass

**form = None**

the *IndicoForm* to use for the request form

**form\_defaults = {}**

default values to use if there's no existing request

**classmethod get\_manager\_notification\_emails** ()

Return the email addresses of users who manage requests of this type.

The email addresses are used only for notifications. It usually makes sense to return the email addresses of the users who pass the **:method:'can\_be\_managed'** check.

**Returns** set of email addresses

**classmethod get\_notification\_reply\_email** ()

Return the *Reply-To* e-mail address for notifications.

**classmethod get\_notification\_template** (*name*, *\*\*context*)

Get the template module for a notification email.

**Parameters**

- **name** – the template name
- **context** – data passed to the template

**manager\_form**

the *IndicoForm* to use for the request manager form

alias of *RequestManagerForm*

**classmethod manager\_save** (*req*, *data*)

Save management-specific data.

This method is called when the management form is submitted without accepting/rejecting the request (which is guaranteed to be already accepted or rejected).

**Parameters**

- **req** – the *Request* of the request
- **data** – the form data from the management form

**name = None**  
the unique internal name of the request type

**plugin = None**  
the plugin containing this request definition - assigned automatically

**classmethod reject** (*req, data, user*)  
Reject the request.

To ensure that additional data is saved, this method should call **:method:'manager\_save'**.

#### Parameters

- **req** – the Request of the request
- **data** – the form data from the management form
- **user** – the user processing the request

**classmethod render\_form** (*event, \*\*kwargs*)  
Render the request form.

#### Parameters

- **event** – the event the request is for
- **kwargs** – arguments passed to the template

**classmethod send** (*req, data*)  
Send a new/modified request.

#### Parameters

- **req** – the Request of the request
- **data** – the form data from the request form

**title = None**  
the title of the request type as shown to users

**classmethod withdraw** (*req, notify\_event\_managers=True*)  
Withdraw the request.

#### Parameters

- **req** – the Request of the request
- **notify\_event\_managers** – if event managers should be notified

## 7.1.16 Session

---

**Todo:** Docstrings (module, models, operations, utilities)

---

### Models

```
class indico.modules.events.sessions.models.sessions.Session (**kwargs)
    Bases: indico.core.db.sqlalchemy.descriptions.DescriptionMixin, indico.core.
    db.sqlalchemy.colors.ColorMixin, indico.core.db.sqlalchemy.protection.
    ProtectionManagersMixin, indico.core.db.sqlalchemy.locations.LocationMixin,
```

```
indico.core.db.sqlalchemy.attachments.AttachedItemsMixin, indico.core.db.
sqlalchemy.notes.AttachedNotesMixin, sqlalchemy.orm.decl_api.Model

ATTACHMENT_FOLDER_ID_COLUMN = 'session_id'

PRELOAD_EVENT_ATTACHED_ITEMS = True

PRELOAD_EVENT_NOTES = True

access_key = None

acl_entries

allow_relationship_preloading = True

background_color

blocks

can_manage_blocks (user, allow_admin=True)
    Check whether a user can manage session blocks.

    This only applies to the blocks themselves, not to contributions inside them.

can_manage_contributions (user, allow_admin=True)
    Check whether a user can manage contributions within the session.

code

conveners

default_colors = ColorTuple(text='202020', background='e3f2d3')

default_contribution_duration

default_render_mode = 2

disallowed_protection_modes = frozenset()

end_dt

event

event_id

friendly_id
    The human-friendly ID for the session

get_non_inheriting_objects ()
    Get a set of child objects that do not inherit protection.

id

inherit_location

inheriting_have_acl = True

is_deleted

is_poster

location_backref_name = 'sessions'

location_parent
    The parent object to consult if the location is inherited.
```

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**note**

**own\_address**

**own\_no\_access\_contact** = None

**own\_room**

**own\_room\_id**

**own\_room\_name**

**own\_venue**

**own\_venue\_id**

**own\_venue\_name**

**possible\_render\_modes** = {<RenderMode.markdown: 2>}

**classmethod preload\_acl\_entries** (*event*)

**protection\_mode**

**protection\_parent**

The parent object to consult for ProtectionMode.inheriting.

**render\_mode** = 2

**session**

Convenience property so all event entities have it.

**start\_dt**

**text\_color**

**title**

**type**

**type\_id**

```
class indico.modules.events.sessions.models.blocks.SessionBlock (**kwargs)
    Bases: indico.core.db.sqlalchemy.locations.LocationMixin, sqlalchemy.orm.
    decl_api.Model
```

`can_access` (*user*, *allow\_admin=True*)  
`can_edit_note` (*user*)  
`can_manage` (*user*, *allow\_admin=True*)  
`can_manage_attachments` (*user*)  
`code`  
`contribution_count`  
`duration`  
`end_dt`  
`event`  
`full_title`  
`has_note`  
`id`  
`inherit_location`  
`location_backref_name` = `'session_blocks'`  
`location_parent`

The parent object to consult if the location is inherited.

`locator`

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

`note`  
`own_address`  
`own_room`  
`own_room_id`  
`own_room_name`  
`own_venue`  
`own_venue_id`  
`own_venue_name`

**person\_links**

Persons associated with this session block

**session\_id****slug****start\_dt****title**

```
class indico.modules.events.sessions.models.persons.SessionBlockPersonLink(*args,
                                                                           **kwargs)
```

Bases: *indico.modules.events.models.persons.PersonLinkBase*

Association between EventPerson and SessionBlock.

Also known as a ‘session convener’.

**display\_order****id**

**object\_relationship\_name** = 'session\_block'

**person****person\_id**

**person\_link\_backref\_name** = 'session\_block\_links'

**person\_link\_unique\_columns** = ('session\_block\_id',)

**session\_block\_id**

```
class indico.modules.events.sessions.models.principals.SessionPrincipal(**kwargs)
Bases: indico.core.db.sqlalchemy.principals.PrincipalPermissionsMixin,
       sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

**allow\_category\_roles** = True

**allow\_emails** = True

**allow\_event\_roles** = True

**allow\_registration\_forms** = True

**category\_role****category\_role\_id**

**disallowed\_protection\_modes** = frozenset()

**email****event\_role****event\_role\_id****full\_access****id**

The ID of the acl entry

```
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
permissions
principal_backref_name = 'in_session_acls'
principal_for = 'Session'
read_access
registration_form
registration_form_id
session_id
    The ID of the associated session
type
unique_columns = ('session_id',)
user
user_id
```

## Operations

```
indico.modules.events.sessions.operations.create_session(event, data)
    Create a new session with the information passed in the data argument.

indico.modules.events.sessions.operations.create_session_block(session_, data)

indico.modules.events.sessions.operations.delete_session(event_session)
    Delete session from the event.

indico.modules.events.sessions.operations.delete_session_block(session_block)

indico.modules.events.sessions.operations.update_session(event_session, data)
    Update a session based on the information in the data.

indico.modules.events.sessions.operations.update_session_block(session_block,
                                                                data)
    Update a session block with data passed in the data argument.

indico.modules.events.sessions.operations.update_session_coordinator_privs(event,
                                                                data)
```

## Utilities

```
class indico.modules.events.sessions.util.SessionListToPDF(sessions)
    Bases: indico.legacy.pdfinterface.base.PDFBase

    getBody(story=None)
        Add the content to the story.
```



`indico.modules.events.sessions.util.can_manage_sessions` (*user*, *event*, *permission=None*)

Check whether a user can manage any sessions in an event.

`indico.modules.events.sessions.util.generate_pdf_from_sessions` (*sessions*)

Generate a PDF file from a given session list.

`indico.modules.events.sessions.util.generate_spreadsheet_from_sessions` (*sessions*)

Generate spreadsheet data from a given session list.

**Parameters** *sessions* – The sessions to include in the spreadsheet

`indico.modules.events.sessions.util.get_events_with_linked_sessions` (*user*, *dt=None*)

Return a dict with keys representing event\_id and the values containing data about the user rights for sessions within the event.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

`indico.modules.events.sessions.util.get_session_timetable_pdf` (*sess*, *\*\*kwargs*)

`indico.modules.events.sessions.util.get_sessions_for_user` (*event*, *user*)

`indico.modules.events.sessions.util.has_sessions_for_user` (*event*, *user*)

`indico.modules.events.sessions.util.render_session_type_row` (*session\_type*)

`indico.modules.events.sessions.util.serialize_session_for_ical` (*sess*)

`indico.modules.events.sessions.util.session_coordinator_priv_enabled` (*event*, *priv*)

Check whether a coordinator privilege is enabled.

Currently the following privileges are available:

- manage-contributions
- manage-blocks

**Parameters**

- **event** – The *Event* to check for
- **priv** – The name of the privilege

## 7.1.17 Survey

---

**Todo:** Docstrings (module, models)

---

### Models

**class** `indico.modules.events.surveys.models.surveys.Survey` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**anonymous**

Whether submissions will not be linked to a user

**can\_submit** (*user*)**close** ()**end\_dt**

Datetime when the survey is closed

**event**

The Event containing this survey

**event\_id**

The ID of the event

**has\_ended****has\_started****id**

The ID of the survey

**introduction****is\_active****is\_deleted**

Whether the survey has been marked as deleted

**is\_visible****items**

The list of items

**limit\_reached****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**new\_submission\_emails**

Email addresses to notify about new submissions

**notifications\_enabled**

Whether to send survey related notifications to users

**notify\_participants**

Whether include Participants / Registrants when sending start notifications

**open()****partial\_completion**

Whether answers can be saved without submitting the survey

**private****questions**

The list of questions

**require\_user**

Whether submissions must be done by logged users

**sections**

The list of sections

**send\_start\_notification()****send\_submission\_notification** (*submission*)**start\_dt**

Datetime when the survey is open

**start\_notification\_emails**

Email addresses to notify about the start of a survey

**start\_notification\_recipients**

Return all recipients of the notifications.

This includes both explicit recipients and, if enabled, participants of the event.

**start\_notification\_sent**

Whether start notification has been already sent

**state****submission\_limit**

Maximum number of submissions allowed

**submissions**

The list of submissions

**title**

The title of the survey

**uuid**

**class** indico.modules.events-surveys.models-surveys.SurveyState

Bases: indico.util.enum.IndicoEnum

An enumeration.

**active\_and\_answered** = 4

**active\_and\_clean** = 3

**finished** = 5

**limit\_reached** = 6

**not\_ready** = 1

**ready\_to\_open = 2**

**class** indico.modules.events.surveys.models.items.**SurveyItem**(\*\*kwargs)  
Bases: indico.core.db.sqlalchemy.descriptions.DescriptionMixin, sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**default\_render\_mode = 2**

**display\_as\_section**

If a section should be rendered as a section

**field\_data**

Field-specific data (such as choices for multi-select fields)

**field\_type**

The type of the field used for the question

**id**

The ID of the item

**is\_required**

If the question must be answered (wtforms DataRequired)

**parent\_id**

The ID of the parent section item (NULL for top-level items, i.e. sections)

**position**

The position of the item in the survey form

**possible\_render\_modes = {<RenderMode.markdown: 2>}**

**render\_mode = 2**

**survey\_id**

The ID of the survey

**title**

The title of the item

**to\_dict()**

Return a json-serializable representation of this object.

Subclasses must add their own data to the dict.

**type**

The type of the survey item

**class** indico.modules.events.surveys.models.items.**SurveyItemType**

Bases: int, indico.util.enum.IndicoEnum

An enumeration.

**question = 1**

**section = 2**

**text = 3**

---

```
class indico.modules.events.surveys.models.items.SurveyQuestion(**kwargs)
```

Bases: *indico.modules.events.surveys.models.items.SurveyItem*

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**display\_as\_section**

**field**

**field\_data**

**field\_type**

```
get_summary(**kwargs)
```

Return the summary of answers submitted for this question.

**id**

**is\_required**

**locator**

**not\_empty\_answers**

**parent\_id**

**position**

**render\_mode = 2**

**survey\_id**

**title**

```
to_dict()
```

Return a json-serializable representation of this object.

Subclasses must add their own data to the dict.

**type**

```
class indico.modules.events.surveys.models.items.SurveySection(**kwargs)
```

Bases: *indico.modules.events.surveys.models.items.SurveyItem*

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**children**

The child items of this section

**display\_as\_section**

**field\_data**

**field\_type**

**id**

**is\_required**

**locator**  
**parent\_id**  
**position**  
**render\_mode** = 2  
**survey\_id**  
**title**  
**to\_dict()**  
Return a json-serializable representation of this object.  
Subclasses must add their own data to the dict.

**type**

**class** `indico.modules.events.surveys.models.items.SurveyText` (\*\*kwargs)  
Bases: `indico.modules.events.surveys.models.items.SurveyItem`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**display\_as\_section**

**field\_data**

**field\_type**

**id**

**is\_required**

**locator**

**parent\_id**

**position**

**render\_mode** = 2

**survey\_id**

**title**

**to\_dict()**  
Return a json-serializable representation of this object.  
Subclasses must add their own data to the dict.

**type**

**class** `indico.modules.events.surveys.models.submissions.SurveyAnswer` (\*\*kwargs)  
Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**answer\_data**

**data**  
The user's answer (no, not 42!) to the question

**is\_empty**

**question**  
The list of answers

**question\_id**  
The ID of the question

**submission\_id**  
The ID of the submission

```
class indico.modules.events-surveys.models-submissions.SurveySubmission (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**answers**  
The list of answers

**friendly\_id**  
The human-friendly ID of the submission

**id**  
The ID of the submission

**is\_anonymous**  
Whether the survey submission is anonymous

**is\_submitted**  
Whether the survey was submitted

**locator**

**pending\_answers**  
List of non-submitted answers

**submitted\_dt**  
The date/time when the survey was submitted

**survey\_id**  
The ID of the survey

**user**  
The user who submitted the survey

**user\_id**  
The ID of the user who submitted the survey

## Operations

```
indico.modules.events-surveys.operations.add_survey_question (section, field_cls,
                                                                data)
```

Add a question to a survey.

### Parameters

- **section** – The *SurveySection* to which the question will be added.
- **field\_cls** – The field class of this question.
- **data** – The *FieldConfigForm.data* to populate the question with.

**Returns** The added *SurveyQuestion*.

`indico.modules.events.surveys.operations.add_survey_section(survey, data)`  
Add a section to a survey.

**Parameters**

- **survey** – The *Survey* to which the section will be added.
- **data** – Attributes of the new *SurveySection*.

**Returns** The added *SurveySection*.

`indico.modules.events.surveys.operations.add_survey_text(section, data)`  
Add a text item to a survey.

**Parameters**

- **section** – The *SurveySection* to which the question will be added.
- **data** – The *TextForm.data* to populate the question with.

**Returns** The added *SurveyText*.

## Utilities

`indico.modules.events.surveys.util.generate_spreadsheet_from_survey(survey, submission_ids)`

Generate spreadsheet data from a given survey.

**Parameters**

- **survey** – *Survey* for which the user wants to export submissions
- **submission\_ids** – The list of submissions to include in the file

`indico.modules.events.surveys.util.get_events_with_submitted_surveys(user, dt=None)`

Get the IDs of events where the user submitted a survey.

**Parameters**

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

**Returns** A set of event ids

`indico.modules.events.surveys.util.is_submission_in_progress(survey)`  
Check whether the current user has a survey submission in progress.

`indico.modules.events.surveys.util.make_survey_form(survey)`  
Create a WTForm from survey questions.

Each question will use a field named `question_ID`.

**Parameters** **survey** – The *Survey* for which to create the form.

**Returns** An *IndicoForm* subclass.



`indico.modules.events.surveys.util.query_active_surveys(event)`

`indico.modules.events.surveys.util.save_submitted_survey_to_session(submission)`  
 Save submission of a survey to session for further checks.

`indico.modules.events.surveys.util.was_survey_submitted(survey)`  
 Check whether the current user has submitted a survey.

## 7.1.18 Timetable

---

**Todo:** Docstring (module, models, operations, utilities)

---

### Models

```
class indico.modules.events.timetable.models.breaks.Break (**kwargs)
    Bases:      indico.core.db.sqlalchemy.descriptions.DescriptionMixin, indico.
               core.db.sqlalchemy.colors.ColorMixin, indico.core.db.sqlalchemy.locations.
               LocationMixin, sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**background\_color**

**can\_access** (user)

**default\_colors** = ColorTuple(text='202020', background='90c0f0')

**default\_render\_mode** = 2

**duration**

**end\_dt**

**event**

**id**

**inherit\_location**

**location\_backref\_name** = 'breaks'

**location\_parent**

The parent object to consult if the location is inherited.

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**own\_address**

**own\_room**

**own\_room\_id**

**own\_room\_name**

**own\_venue**

**own\_venue\_id**

**own\_venue\_name**

**possible\_render\_modes** = {<RenderMode.markdown: 2>}

**render\_mode** = 2

**start\_dt**

**text\_color**

**title**

```
class indico.modules.events.timetable.models.entries.TimetableEntry(**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**break\_**

**break\_id**

**can\_view** (*user*)

Check whether the user will see this entry in the timetable.

**children**

**contribution**

**contribution\_id**

**duration**

**end\_dt**

**event**

**event\_id**

**extend\_end\_dt** (*end\_dt*)

**extend\_parent** (*by\_start=True, by\_end=True*)

Extend start/end of parent objects if needed.

No extension if performed for entries crossing a day boundary in the event timezone.

#### Parameters

- **by\_start** – Extend parent by start datetime.
- **by\_end** – Extend parent by end datetime.

**extend\_start\_dt** (*start\_dt*)

**id**

**is\_parallel** (*in\_session=False*)

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**move** (*start\_dt*)

Move the entry to start at a different time.

This method automatically moves children of the entry to preserve their start time relative to the parent's start time.

**move\_next\_to** (*sibling, position='before'*)

**object**

**parent\_id**

**session\_block**

**session\_block\_id**

**session\_siblings**

**siblings**

**siblings\_query**

**start\_dt**

**type**

```
class indico.modules.events.timetable.models.entries.TimetableEntryType
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    BREAK = 3
    CONTRIBUTION = 2
    SESSION_BLOCK = 1
```

## Operations

```
indico.modules.events.timetable.operations.can_swap_entry(entry,          direction,
                                                           in_session=False)
indico.modules.events.timetable.operations.create_break_entry(event, data, ses-
                                                                sion_block=None)
indico.modules.events.timetable.operations.create_session_block_entry(session_,
                                                                        data)
indico.modules.events.timetable.operations.create_timetable_entry(event,
                                                                    data, par-
                                                                    ent=None,
                                                                    ex-
                                                                    tend_parent=False)
indico.modules.events.timetable.operations.delete_timetable_entry(entry,
                                                                    log=True)
indico.modules.events.timetable.operations.fit_session_block_entry(entry,
                                                                    log=True)
indico.modules.events.timetable.operations.get_sibling_entry(entry,    direction,
                                                                in_session=False)
indico.modules.events.timetable.operations.move_timetable_entry(entry,    par-
                                                                ent=None,
                                                                day=None)

    Move the entry to another session or top-level timetable.
```

### Parameters

- **entry** – *TimetableEntry* to be moved
- **parent** – If specified then the entry will be set as a child of parent
- **day** – If specified then the entry will be moved to the top-level timetable on this day

```
indico.modules.events.timetable.operations.schedule_contribution(contribution,
                                                                    start_dt, ses-
                                                                    sion_block=None,
                                                                    ex-
                                                                    tend_parent=False)
indico.modules.events.timetable.operations.swap_timetable_entry(entry,    di-
                                                                rection,    ses-
                                                                sion_=None)
```

Swap entry with closest gap or non-parallel sibling.

```
indico.modules.events.timetable.operations.update_break_entry(break_, data)
indico.modules.events.timetable.operations.update_timetable_entry(entry,
                                                                    data)
```

`indico.modules.events.timetable.operations.update_timetable_entry_object` (*entry*, *data*)  
 Update the *object* of a timetable entry according to its type.

## Utilities

`indico.modules.events.timetable.util.find_latest_entry_end_dt` (*obj*, *day=None*)  
 Get the latest end datetime for timetable entries within the object.

### Parameters

- **obj** – The Event or SessionBlock that will be used to look for timetable entries.
- **day** – The local event date to look for timetable entries. Applicable only to Event.

**Returns** The end datetime of the timetable entry finishing the latest. None if no entry was found.

`indico.modules.events.timetable.util.find_next_start_dt` (*duration*, *obj*, *day=None*, *force=False*)

Find the next most convenient start date fitting a duration within an object.

### Parameters

- **duration** – Duration to fit into the event/session-block.
- **obj** – The Event or SessionBlock the duration needs to fit into.
- **day** – The local event date where to fit the duration in case the object is an event.
- **force** – Gives earliest datetime if the duration doesn't fit.

**Returns** The end datetime of the latest scheduled entry in the object if the duration fits then. If it doesn't, the latest datetime that fits it. None if the duration cannot fit in the object, earliest datetime if *force* is True.

`indico.modules.events.timetable.util.get_category_timetable` (*categ\_ids*, *start\_dt*, *end\_dt*, *detail\_level='event'*, *tz=<UTC>*, *from\_categ=None*, *grouped=True*, *includible=<function <lambda>>*)

Retrieve time blocks that fall within a specific time interval for a given set of categories.

### Parameters

- **categ\_ids** – iterable containing list of category IDs
- **start\_dt** – start of search interval (datetime, expected to be in display timezone)
- **end\_dt** – end of search interval (datetime in expected to be in display timezone)
- **detail\_level** – the level of detail of information (event|session|contribution)
- **tz** – the timezone information should be displayed in
- **from\_categ** – Category that will be taken into account to calculate visibility
- **grouped** – Whether to group results by start date

- **includible** – a callable, to allow further arbitrary custom filtering (maybe from 3rd party plugins) on whether to include (returns True) or not (returns False) each detail item. Default always returns True.

**Returns** a dictionary containing timetable information in a structured way. See source code for examples.

```
indico.modules.events.timetable.util.get_nested_entries(event)
```

```
indico.modules.events.timetable.util.get_session_block_entries(event, day)
```

Return a list of event top-level session blocks for the given *day*.

```
indico.modules.events.timetable.util.get_time_changes_notifications(changes,
                                                                    tzinfo,
                                                                    en-
                                                                    try=None)
```

```
indico.modules.events.timetable.util.get_timetable_offline_pdf_generator(event)
```

```
indico.modules.events.timetable.util.get_top_level_entries(event)
```

```
indico.modules.events.timetable.util.render_entry_info_balloon(entry,      ed-
                                                                itable=False,
                                                                sess=None,
                                                                is_session_timetable=False)
```

```
indico.modules.events.timetable.util.render_session_timetable(session,
                                                                timetable_layout=None,
                                                                manage-
                                                                ment=False)
```

```
indico.modules.events.timetable.util.shift_following_entries(entry, shift, ses-
                                                                sion_=None)
```

Reschedule entries starting after the given entry by the given shift.

```
class indico.modules.events.timetable.reschedule.RescheduleMode
```

Bases: `str`, `indico.util.enum.RichEnum`

An enumeration.

```
duration = 'duration'
```

```
none = 'none'
```

```
time = 'time'
```

```
class indico.modules.events.timetable.reschedule.Rescheduler(event, mode, day,
                                                                session=None, ses-
                                                                sion_block=None,
                                                                fit_blocks=False,
                                                                gap=datetime.timedelta(0))
```

Bases: `object`

Compact the the schedule of an event day by either adjusting start times or durations of timetable entries.

#### Parameters

- **event** – The event of which the timetable entries should be rescheduled.
- **mode** – A *RescheduleMode* value specifying whether the duration or start time should be adjusted.
- **day** – A *date* specifying the day to reschedule (the day of the timetable entries are determined using the event's timezone)

- **session** – If specified, only blocks of that session will be rescheduled, ignoring any other timetable entries. Cannot be combined with *session\_block*.
- **session\_block`** – If specified, only entries inside that block will be rescheduled. Cannot be combined with *session*.
- **fit\_blocks** – Whether session blocks should be resized to exactly fit their contents before the actual rescheduling operation.
- **gap** – A timedelta specifying the cap between rescheduled timetable entries.

**run()**

Perform the rescheduling.

## 7.1.19 Track

---

**Todo:** Docstring (module, models, operations)

---

### Models

```
class indico.modules.events.tracks.models.tracks.Track (**kwargs)
    Bases: indico.core.db.sqlalchemy.descriptions.DescriptionMixin, indico.core.db.sqlalchemy.protection.ProtectionManagersMixin, sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
access_key = None
acl_entries
can_convene (user)
can_delete (user)
can_review_abstracts (user)
code
default_render_mode = 2
default_session
default_session_id
disable_protection_mode = True
event
event_id
full_title
full_title_with_group
id
```

```
is_track_group = False
```

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```
own_no_access_contact = None
```

**position**

```
possible_render_modes = {<RenderMode.markdown: 2>}
```

**protection\_mode** = None

**render\_mode** = 2

**short\_title**

**short\_title\_with\_group**

**title**

**title\_with\_group**

**track\_group**

**track\_group\_id**

```
indico.modules.events.tracks.models.tracks.get_next_position(context)
```

```
class indico.modules.events.tracks.models.principals.TrackPrincipal(**kwargs)
    Bases: indico.core.db.sqlalchemy.principals.PrincipalPermissionsMixin,
           sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**allow\_category\_roles** = True

**allow\_event\_roles** = True

**category\_role**

**category\_role\_id**



```

email = None
event_role
event_role_id
full_access
id
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
permissions
principal_backref_name = 'in_track_acls'
principal_for = 'Track'
read_access
registration_form = None
registration_form_id = None
track_id
type
unique_columns = ('track_id',)
user
user_id

```

## Operations

```

indico.modules.events.tracks.operations.create_track(event, data)
indico.modules.events.tracks.operations.create_track_group(event, data)
indico.modules.events.tracks.operations.delete_track(track)
indico.modules.events.tracks.operations.delete_track_group(track_group)
indico.modules.events.tracks.operations.update_program(event, data)
indico.modules.events.tracks.operations.update_track(track, data)
indico.modules.events.tracks.operations.update_track_group(track_group, data)

```

### 7.1.20 Static site

---

**Todo:** Doctrings (module, utilities)

---

## Models

```
class indico.modules.events.static.models.static.StaticSite(**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model

    Static site for an Indico event.

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

add_file_date_column = False

content_type
    The MIME type of the file.

created_dt = None

creator
    The user who created the static site

creator_id
    ID of the user who created the static site

event
    The Event this static site is associated with

event_id
    ID of the event

extension
    The extension of the file.

file_required = False

filename
    The name of the file.

id
    Entry ID

locator

md5
    An MD5 hash of the file.

    Automatically assigned when save() is called.

requested_dt
    The date and time the static site was requested

size
    The size of the file (in bytes).

    Automatically assigned when save() is called.

state
    The state of the static site (a StaticSiteState member)

storage_backend

storage_file_id
```

```
class indico.modules.events.static.models.static.StaticSiteState
    Bases: indico.util.enum.RichIntEnum

    An enumeration.

    expired = 4
    failed = 3
    pending = 0
    running = 1
    success = 2
```

## Utilities

```
class indico.modules.events.static.util.RewrittenManifest (manifest)
    Bases: pywebpack.manifests.Manifest

    A manifest that rewrites its asset paths.

indico.modules.events.static.util.collect_static_files()
    Keep track of URLs used by manifest and url_for.

indico.modules.events.static.util.override_request_endpoint (endpoint)

indico.modules.events.static.util.rewrite_css_urls (event, css)
    Rewrite CSS in order to handle url(...) properly.

indico.modules.events.static.util.rewrite_static_url (path)
    Remove __vxxx prefix from static URLs.

indico.modules.events.static.util.url_to_static_filename (endpoint, url)
    Handle special endpoint/URLs so that they link to offline content.
```

## 7.1.21 Category

---

**Todo:** Docstrings (module, model, operations, utilities)

---

## Models

```
class indico.modules.categories.models.categories.Category (**kwargs)
    Bases: indico.core.db.sqlalchemy.searchable.SearchableTitleMixin, indico.
            core.db.sqlalchemy.descriptions.DescriptionMixin, indico.core.db.
            sqlalchemy.protection.ProtectionManagersMixin, indico.core.db.sqlalchemy.
            attachments.AttachedItemsMixin, sqlalchemy.orm.decl_api.Model

    An Indico category.

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    ATTACHMENT_FOLDER_ID_COLUMN = 'category_id'
```

**access\_key** = None

**acl\_entries**

**allow\_no\_access\_contact** = True

**allow\_relationship\_preloading** = True

**can\_create\_events** (*user*)  
Check whether the user can create events in the category.

**chain\_query**  
Get a query object for the category chain.  
  
The query retrieves the root category first and then all the intermediate categories up to (and including) this category.

**children**

**deep\_children\_query**  
Get a query object for all subcategories.  
  
This includes subcategories at any level of nesting.

**default\_badge\_template**

**default\_badge\_template\_id**

**default\_event\_themes**

**default\_render\_mode** = 2

**default\_ticket\_template**

**default\_ticket\_template\_id**

**disallowed\_protection\_modes** = frozenset()

**display\_tzinfo**  
The tzinfo of the category or the one specified by the user.

**effective\_icon\_url**  
Get the HTTP URL of the icon (possibly inherited).

**event\_creation\_notification\_emails**

**event\_creation\_restricted**

**event\_message**

**event\_message\_mode**

**get\_hidden\_events** (*user=None*)  
Get all hidden events within the given category and user.

**classmethod get\_icon\_data\_cte** ()

**classmethod get\_protection\_cte** ()

**get\_protection\_parent\_cte** ()

**classmethod get\_root** ()  
Get the root category.

**classmethod get\_tree\_cte** (*col='id'*)  
Create a CTE for the category tree.  
  
The CTE contains the following columns:

- `id` – the category id
- **`path`** – an array containing the path from the root to the category itself
- `is_deleted` – whether the category is deleted

**Parameters** `col` – The name of the column to use in the path or a callable receiving the category alias that must return the expression used for the ‘path’ retrieved by the CTE.

**static** `get_visible_categories_cte(category_id)`

Get a sqlalchemy select for the visible categories within the given category, including the category itself.

`has_effective_icon`

`has_icon`

`has_logo`

`icon`

`icon_metadata`

`icon_url`

Get the HTTP URL of the icon.

`id`

`inheriting_have_acl = True`

`is_deleted`

`is_descendant_of(categ)`

`is_empty`

`is_flat_view_enabled`

`is_root`

`locator`

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

`logo`

`logo_metadata`

**logo\_url**

Get the HTTP URL of the logo.

**move** (*target*)

Move the category into another category.

**notify\_managers**

**nth\_parent** (*n\_cats*, *fail\_on\_overflow=True*)

Return the *n*th parent of the category.

**Parameters**

- **n\_cats** – the number of categories to go up
- **fail\_on\_overflow** – whether to fail if we try to go above the root category

**Returns** *Category* object or None (only if *fail\_on\_overflow* is not set)

**own\_no\_access\_contact**

**own\_visibility\_horizon**

Get the highest category this one would like to be visible from (configured visibility).

**parent\_chain\_query**

Get a query object for the category's parent chain.

The query retrieves the root category first and then all the intermediate categories up to (excluding) this category.

**parent\_id**

**position**

**possible\_render\_modes** = {<RenderMode.markdown: 2>}

**protection\_mode**

**protection\_parent**

The parent object to consult for ProtectionMode.inheriting.

**real\_visibility\_horizon**

Get the highest category this one is actually visible from (as limited by categories above).

**render\_mode** = 2

**suggestions\_disabled**

**timezone**

**title**

**tzinfo**

**url**

**visibility**

**visibility\_horizon\_query**

Get a query object that returns the highest category this one is visible from.

**visible\_categories\_query**

Get a query object for the visible categories within this category, including the category itself.

**class** `indico.modules.categories.models.categories.EventMessageMode`

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

```

danger = 3
disabled = 0
info = 1
warning = 2

```

```

class indico.modules.categories.models.principals.CategoryPrincipal(**kwargs)
    Bases: indico.core.db.sqlalchemy.principals.PrincipalPermissionsMixin,
           sqlalchemy.orm.decl_api.Model

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

allow_category_roles = True
allow_networks = True
category_id
    The ID of the associated event
category_role
category_role_id
email = None
event_role = None
event_role_id = None
full_access
id
    The ID of the acl entry
ip_network_group
ip_network_group_id
local_group
local_group_id
multipass_group_name
multipass_group_provider
permissions
principal_backref_name = 'in_category_acls'
principal_for = 'Category'
read_access
registration_form = None
registration_form_id = None
type
unique_columns = ('category_id',)
user

```

**user\_id**

```
class indico.modules.categories.models.settings.CategorySetting (**kwargs)
    Bases: indico.core.settings.models.base.JSONSettingsBase, sqlalchemy.orm.
    decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**category**

**category\_id**

**id**

**module**

**name**

**value**

## Operations

```
indico.modules.categories.operations.create_category (parent, data)
```

```
indico.modules.categories.operations.delete_category (category)
```

```
indico.modules.categories.operations.move_category (category, target_category)
```

```
indico.modules.categories.operations.update_category (category, data, skip=())
```

## Utilities

```
indico.modules.categories.util.get_attachment_count (category_id=None)
```

Get the number of attachments in events in a category.

**Parameters** **category\_id** – The category ID to get statistics for. Attachments from subcategories are also included.

**Returns** The number of attachments

```
indico.modules.categories.util.get_category_stats (category_id=None)
```

Get category statistics.

This function is mainly a helper so we can get and cache all values at once and keep a last-update timestamp.

**Parameters** **category\_id** – The category ID to get statistics for. Subcategories are also included.

```
indico.modules.categories.util.get_contribs_by_year (category_id=None)
```

Get the number of contributions for each year.

**Parameters** **category\_id** – The category ID to get statistics for. Contributions from subcategories are also included.

**Returns** A dictionary mapping years to contribution counts.

```
indico.modules.categories.util.get_events_by_year (category_id=None)
```

Get the number of events for each year.



**Parameters** **category\_id** – The category ID to get statistics for. Events from subcategories are also included.

**Returns** A dictionary mapping years to event counts.

```
indico.modules.categories.util.get_image_data(image_type, category)
```

```
indico.modules.categories.util.get_min_year(category_id=None)
```

Get the min year.

**Parameters** **category\_id** – The category ID to get statistics for.

**Returns** The year.

```
indico.modules.categories.util.get_upcoming_events()
```

Get the global list of upcoming events.

```
indico.modules.categories.util.get_visibility_options(category_or_event, al-
low_invisible=True)
```

Return the visibility options available for the category or event.

```
indico.modules.categories.util.serialize_category_role(role, legacy=True)
```

Serialize role to JSON-like object.

```
indico.modules.categories.serialize.serialize_categories_ical(category_ids,
user,
event_filter=True,
event_filter_fn=None,
up-
date_query=None)
```

Export the events in a category to iCal.

#### Parameters

- **category\_ids** – Category IDs to export
- **user** – The user who needs to be able to access the events
- **event\_filter** – A SQLAlchemy criterion to restrict which events will be returned. Usually something involving the start/end date of the event.
- **event\_filter\_fn** – A callable that determines which events to include (after querying)
- **update\_query** – A callable that can update the query used to retrieve the events. Must return the updated query object.

```
indico.modules.categories.serialize.serialize_category(category,
with_favorite=False,
with_path=False,
parent_path=None,
child_path=None)
```

```
indico.modules.categories.serialize.serialize_category_atom(category, url, user,
event_filter)
```

Export the events in a category to Atom.

#### Parameters

- **category** – The category to export
- **url** – The URL of the feed
- **user** – The user who needs to be able to access the events
- **event\_filter** – A SQLAlchemy criterion to restrict which events will be returned. Usually something involving the start/end date of the event.

```
indico.modules.categories.serialize.serialize_category_chain(category, include_children=False, include_parents=False)
```

## Settings

```
class indico.modules.categories.settings.CategorySettingsProxy(module, defaults=None, strict=True, acls=None, converters=None)
```

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access category-specific settings for a certain module.

**delete** (*category*, \**names*)  
Delete settings.

### Parameters

- **category** – Category (or its ID)
- **names** – One or more names of settings to delete

**delete\_all** (*category*)  
Delete all settings.

**Parameters** **category** – Category (or its ID)

**get** (*category*, *name*, *default*=<object object>)  
Retrieve the value of a single setting.

### Parameters

- **category** – Category (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

**Returns** The settings's value or the default value

**get\_all** (*category*, *no\_defaults*=False)  
Retrieve all settings.

### Parameters

- **category** – Category (or its ID)
- **no\_defaults** – Only return existing settings and ignore defaults.

**Returns** Dict containing the settings

**query**  
Return a query object filtering by the proxy's module.

**set** (*category*, *name*, *value*)  
Set a single setting.

### Parameters

- **category** – Category (or its ID)
- **name** – Setting name

- **value** – Setting value; must be JSON-serializable

**set\_multi** (*category, items*)  
Set multiple settings at once.

#### Parameters

- **category** – Category (or its ID)
- **items** – Dict containing the new settings

## 7.1.22 User

---

**Todo:** Docstrings (module, models, utilities)

---

### Models

**class** `indico.modules.users.models.users.NameFormat`

Bases: `indico.util.enum.RichIntEnum`

An enumeration.

**f\_last** = 3

**f\_last\_upper** = 7

**first\_last** = 0

**first\_last\_upper** = 4

**last\_f** = 2

**last\_f\_upper** = 6

**last\_first** = 1

**last\_first\_upper** = 5

**class** `indico.modules.users.models.users.PersonMixin`

Bases: `object`

Add convenience properties and methods to person classes.

Assumes the following attributes exist: \* `first_name` \* `last_name` \* `title`

**display\_full\_name**

Return the full name using the user's preferred name format.

**full\_name**

Return the person's name in 'Firstname Lastname' notation.

**get\_full\_name** (*last\_name\_first=True, last\_name\_upper=True, abbrev\_first\_name=True, show\_title=False, \_show\_empty\_names=False*)

Return the person's name in the specified notation.

Note: Do not use positional arguments when calling this method. Always use keyword arguments!

#### Parameters

- **last\_name\_first** – if “lastname, firstname” instead of “firstname lastname” should be used

- **last\_name\_upper** – if the last name should be all-uppercase
- **abbrev\_first\_name** – if the first name should be abbreviated to use only the first character
- **show\_title** – if the title of the person should be included

**name**

Return the person's name in 'Firstname Lastname' notation.

**title**

The title of the user

```
class indico.modules.users.models.users.ProfilePictureSource
```

Bases: `int`, `enum.Enum`

An enumeration.

**custom** = 3

**gravatar** = 2

**identicon** = 1

**standard** = 0

```
class indico.modules.users.models.users.User(**kwargs)
```

Bases: `indico.modules.users.models.users.PersonMixin`, `sqlalchemy.orm.decl_api.Model`

Indico users.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**address**

the address of the user

**affiliation**

the affiliation of the user

**all\_emails**

all emails of the user. read-only; use it only for searching by email! also, do not use it between modifying *email* or *secondary\_emails* and a session expire/commit!

**api\_key**

the active API key of the user

**as\_principal**

The serializable principal identifier of this user.

**avatar\_bg\_color****avatar\_url****can\_be\_modified** (*user*)

If this user can be modified by the given user.

**can\_get\_all\_multipass\_groups**

Check whether it is possible to get all multipass groups the user is in.

**email**

the primary email address of the user

**event\_log\_entries****event\_notes\_revisions****external\_identities**

The external identities of the user.

**favorite\_categories**

the users's favorite categories

**favorite\_users**

the users's favorite users

**first\_name**

the first name of the user

**get\_full\_name** (\*args, \*\*kwargs)

Return the person's name in the specified notation.

Note: Do not use positional arguments when calling this method. Always use keyword arguments!

**Parameters**

- **last\_name\_first** – if “lastname, firstname” instead of “firstname lastname” should be used
- **last\_name\_upper** – if the last name should be all-uppercase
- **abbrev\_first\_name** – if the first name should be abbreviated to use only the first character
- **show\_title** – if the title of the person should be included

**get\_identity** (provider)

Return the first user identity which matches the given provider.

**Parameters** **provider** – The id of the provider in question

**Returns** The requested identity, or *None* if none is found

**get\_merged\_from\_users\_recursive** ()

Get the users merged into this users recursively.

**static get\_system\_user** ()**has\_picture****id**

the unique id of the user

**identifier****identities**

the identities used by this user

**is\_admin**

if the user is an administrator with unrestricted access to everything

**is\_blocked**

if the user has been blocked

**is\_deleted**

if the user is deleted (e.g. due to a merge)

**is\_pending**

if the user is pending (e.g. never logged in, only added to some list)

**is\_system**

if the user is the default system user

**iter\_all\_multipass\_groups()**

Iterate over all multipass groups the user is in.

**iter\_identifiers** (*check\_providers=False, providers=None*)

Yields (provider, identifier) tuples for the user.

**Parameters**

- **check\_providers** – If True, providers are searched for additional identifiers once all existing identifiers have been yielded.
- **providers** – May be a set containing provider names to get only identifiers from the specified providers.

**last\_login\_dt**

The datetime when the user last logged in.

**last\_name**

the last/family name of the user

**local\_identities**

The local identities of the user.

**local\_identity**

The main (most recently used) local identity.

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**make\_email\_primary** (*email*)

Promote a secondary email address to the primary email address.

**Parameters** **email** – an email address that is currently a secondary email

**merged\_into\_id**

the id of the user this user has been merged into

**merged\_into\_user**

the user this user has been merged into

**old\_api\_keys**

the previous API keys of the user

**phone**

the phone number of the user

**picture**

the user profile picture

**picture\_metadata**

user profile picture metadata

**picture\_source**

user profile picture source

**principal\_order = 0****principal\_type = 1****query\_personal\_tokens** (\*, *include\_revoked=False*)

Query the personal tokens of the user.

**Parameters** **include\_revoked** – whether to query revoked tokens as well

**reset\_signing\_secret** ()**secondary\_emails**

any additional emails the user might have

**secondary\_local\_identities**

The local identities of the user except the main one.

**settings**

Return the user settings proxy for this user.

**signing\_secret**

a unique secret used to generate signed URLs

**suggested\_categories**

the user's category suggestions

**synced\_fields**

The fields of the user whose values are currently synced.

This set is always a subset of the synced fields define in synced fields of the idp in 'indico.conf'.

**synced\_values**

The values from the synced identity for the user.

Those values are not the actual user's values and might differ if they are not set as synchronized.

**synchronize\_data** (*refresh=False*, *silent=False*)

Synchronize the fields of the user from the sync identity.

This will take only into account *synced\_fields*.

**Parameters**

- **refresh** – bool – Whether to refresh the synced identity with the sync provider before instead of using the stored data. (Only if the sync provider supports refresh.)
- **silent** – bool – Whether to just synchronize but not flash any messages

**class** indico.modules.users.models.users.**UserTitle**

Bases: indico.util.enum.RichIntEnum

An enumeration.

```
dr = 4
mr = 1
mrs = 3
ms = 2
mx = 6
none = 0
prof = 5
```

```
indico.modules.users.models.users.format_display_full_name(user, obj)
```

```
indico.modules.users.models.users.syncable_fields = {'address': l'address', 'affiliation'
```

Fields which can be synced as keys and a mapping to a more human readable version, used for flashing messages

```
class indico.modules.users.models.affiliations.UserAffiliation(**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
id
    the unique id of the affiliations

name
    the affiliation

user_id
    the id of the associated user
```

```
class indico.modules.users.models.emails.UserEmail(**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
email
    the email address

id
    the unique id of the email address

is_primary
    if the email is the user's primary email

is_user_deleted
    if the user is marked as deleted (e.g. due to a merge). DO NOT use this flag when actually deleting an
    email

user_id
    the id of the associated user
```



```

class indico.modules.users.models.suggestions.SuggestedCategory (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    category
    category_id
    is_ignored

    classmethod merge_users (target, source)
        Merge the suggestions for two users.

        Parameters
        • target – The target user of the merge.
        • source – The user that is being merged into target.

    score
    user_id

class indico.modules.users.models.settings.UserSetting (**kwargs)
    Bases: indico.core.settings.models.base.JSONSettingsBase, sqlalchemy.orm.
    decl_api.Model

    User-specific settings.

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    id
    module
    name
    user
    user_id
    value

class indico.modules.users.models.settings.UserSettingsProxy (module,          de-
                                                                faults=None,
                                                                strict=True,
                                                                acls=None,    con-
                                                                verters=None)

    Bases: indico.core.settings.proxy.SettingsProxyBase

    Proxy class to access user-specific settings for a certain module.

    delete (user, *names)
        Delete settings.

        Parameters

```

- **user** – {'user': user} or {'user\_id': id}
- **names** – One or more names of settings to delete

**delete\_all** (*user*)  
Delete all settings.

**Parameters** **user** – {'user': user} or {'user\_id': id}

**get** (*user, name, default=<object object>*)  
Retrieve the value of a single setting.

**Parameters**

- **user** – {'user': user} or {'user\_id': id}
- **name** – Setting name
- **default** – Default value in case the setting does not exist

**Returns** The settings's value or the default value

**get\_all** (*user, no\_defaults=False*)  
Retrieve all settings.

**Parameters**

- **user** – {'user': user} or {'user\_id': id}
- **no\_defaults** – Only return existing settings and ignore defaults.

**Returns** Dict containing the settings

**query**  
Return a query object filtering by the proxy's module.

**set** (*user, name, value*)  
Set a single setting.

**Parameters**

- **user** – {'user': user} or {'user\_id': id}
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

**set\_multi** (*user, items*)  
Set multiple settings at once.

**Parameters**

- **user** – {'user': user} or {'user\_id': id}
- **items** – Dict containing the new settings

`indico.modules.users.models.settings.user_or_id(f)`

## Operations

`indico.modules.users.operations.create_user(email, data, identity=None, settings=None, other_emails=None, from_moderation=True)`

Create a new user.

This may also convert a pending user to a proper user in case the email address matches such a user.

### Parameters

- **email** – The primary email address of the user.
- **data** – The data used to populate the user.
- **identity** – An *Identity* to associate with the user.
- **settings** – A dict containing user settings.
- **other\_emails** – A set of email addresses that are also used to check for a pending user. They will also be added as secondary emails to the user.
- **from\_moderation** – Whether the user was created through the moderation process or manually by an admin.

### Utilities

```
indico.modules.users.util.build_user_search_query(criteria, exact=False, include_deleted=False, include_pending=False, include_blocked=False, favorites_first=False)
```

```
indico.modules.users.util.get_admin_emails()
```

Get the email addresses of all Indico admins.

```
indico.modules.users.util.get_avatar_url_from_name(first_name)
```

```
indico.modules.users.util.get_color_for_user_id(user_id: Union[int, str])
```

Calculate a unique color for a user based on their id.

**Parameters** **user\_id** – the user ID (int), or otherwise a string (external search results)

```
indico.modules.users.util.get_gravatar_for_user(user, identicon, size=256, lastmod=None)
```

```
indico.modules.users.util.get_linked_events(user, dt, limit=None, load_also=())
```

Get the linked events and the user's roles in them.

### Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date
- **limit** – Max number of events

```
indico.modules.users.util.get_related_categories(user, detailed=True)
```

Get the related categories of a user for the dashboard.

```
indico.modules.users.util.get_suggested_categories(user)
```

Get the suggested categories of a user for the dashboard.

```
indico.modules.users.util.get_user_by_email(email, create_pending=False)
```

Find a user based on his email address.

### Parameters

- **email** – The email address of the user.
- **create\_pending** – If True, this function searches for external users and creates a new pending User in case no existing user was found.

**Returns** A *User* instance or *None* if not exactly one user was found.

`indico.modules.users.util.merge_users(source, target, force=False)`

Merge two users together, unifying all related data.

#### Parameters

- **source** – source user (will be set as deleted)
- **target** – target user (final)

`indico.modules.users.util.search_users(exact=False, include_deleted=False, include_pending=False, include_blocked=False, external=False, allow_system_user=False, **criteria)`

Search for users.

#### Parameters

- **exact** – Indicates if only exact matches should be returned. This is MUCH faster than a non-exact search, especially when searching external users.
- **include\_deleted** – Indicates if also users marked as deleted should be returned.
- **include\_pending** – Indicates if also users who are still pending should be returned.
- **include\_blocked** – Indicates if also users marked as blocked should be returned.
- **external** – Indicates if identity providers should be searched for matching users.
- **allow\_system\_user** – Whether the system user may be returned in the search results.
- **criteria** – A dict containing any of the following keys: name, first\_name, last\_name, email, affiliation, phone, address

**Returns** A set of matching users. If *external* was set, it may contain both `IdentityInfo` objects for external users not yet in Indico and `User` objects for existing users.

`indico.modules.users.util.send_avatar(user)`

`indico.modules.users.util.send_default_avatar(user: Union[indico.modules.users.models.users.User, str, None])`

Send a user's default avatar as an SVG.

**Parameters** **user** – A `User` object, string (external search results, registrations) or `None` (blank avatar)

`indico.modules.users.util.serialize_user(user)`

Serialize user to JSON-like object.

`indico.modules.users.util.set_user_avatar(user, avatar, filename, lastmod=None)`

**class** `indico.modules.users.ext.ExtraUserPreferences(user)`

Bases: `object`

Define additional user preferences.

To use this class, subclass it and override *defaults*, *fields* and *save* to implement your custom logic.

**extend\_defaults** (*defaults*)

Add values to the FormDefaults.

**extend\_form** (*form\_class*)

Create a subclass of the form containing the extra field.

**fields** = {}

a dict containing all the fields that should be added to the user preferences

**classmethod `is_active`** (*user*)  
 Return whether the preferences are available for the given user.

**`load`** ()  
 Return a dict with the current values for the user.

**`process_form_data`** (*data*)  
 Process and save submitted data.  
 This modifies *data* so the core code doesn't receive any extra data it doesn't expect.

**`save`** (*data*)  
 Save the updated settings.

### 7.1.23 Attachment

---

**Todo:** Docstrings (module, models, operations)

---

#### Models

**class** `indico.modules.attachments.models.attachments.Attachment` (*\*\*kwargs*)  
 Bases: `indico.core.db.sqlalchemy.searchable.SearchableTitleMixin`, `indico.core.db.sqlalchemy.protection.ProtectionMixin`, `indico.core.storage.models.VersionedResourceMixin`, `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**`absolute_download_url`**  
 The absolute download url for the attachment.

**`access_key`** = `None`

**`acl`**  
 The ACL of the folder (used for `ProtectionMode.protected`)

**`acl_entries`**

**`all_files`**

**`can_access`** (*user*, *\*args*, *\*\*kwargs*)  
 Check if the user is allowed to access the attachment.  
 This is the case if the user has access to see the attachment or if the user can manage attachments for the linked object.

**`description`**  
 The description of the attachment

**`download_url`**  
 The download url for the attachment.

**`file`**

**`file_id`**

**folder**

The folder containing the attachment

**folder\_id**

The ID of the folder the attachment belongs to

**get\_download\_url** (*absolute=False*)

Return the download url for the attachment.

During static site generation this returns a local URL for the file or the target URL for the link.

**Parameters** **absolute** – If the returned URL should be absolute.

**id**

The ID of the attachment

**is\_deleted**

If the attachment has been deleted

**link\_url**

The target URL for a link attachment

**locator****modified\_dt**

The date/time when the attachment was created/modified

**own\_no\_access\_contact** = None**protection\_mode****protection\_parent**

The parent object to consult for ProtectionMode.inheriting.

**stored\_file\_class**

alias of *AttachmentFile*

**stored\_file\_fkey** = 'attachment\_id'**stored\_file\_table** = 'attachments.files'**title****title\_required** = False**type**

The type of the attachment (file or link)

**user**

The user who created the attachment

**user\_id**

The ID of the user who created the attachment

```
class indico.modules.attachments.models.attachments.AttachmentFile (**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, sqlalchemy.orm.decl_api.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**attachment\_id**  
The ID of the associated attachment

**content\_type**  
The MIME type of the file.

**created\_dt**  
The date/time when the file was uploaded.

**extension**  
The extension of the file.

**filename**  
The name of the file.

**id**  
The ID of the file

**is\_previewable**

**md5**  
An MD5 hash of the file.  
Automatically assigned when *save()* is called.

**size**  
The size of the file (in bytes).  
Automatically assigned when *save()* is called.

**storage\_backend**

**storage\_file\_id**

**user**  
The user who uploaded the file

**user\_id**  
The user who uploaded the file

**version\_of = 'attachment'**

**class** indico.modules.attachments.models.attachments.**AttachmentType**  
Bases: indico.util.enum.RichIntEnum

An enumeration.

**file = 1**

**link = 2**

**class** indico.modules.attachments.models.folders.**AttachmentFolder** (\*\*kwargs)  
Bases: indico.core.db.sqlalchemy.links.LinkMixin, indico.core.db.sqlalchemy.protection.ProtectionMixin, sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**access\_key = None**

**acl**  
The ACL of the folder (used for *ProtectionMode.protected*)

**acl\_entries**

**allowed\_link\_types** = frozenset({<LinkType.category: 1>, <LinkType.event: 2>, <LinkType.contribution: 3>, <LinkType.subcontribution: 4>})

**attachments**

The list of attachments that are not deleted, ordered by name

**can\_access** (*user*, \**args*, \*\**kwargs*)

Check if the user is allowed to access the folder.

This is the case if the user has access the folder or if the user can manage attachments for the linked object.

**can\_view** (*user*)

Check if the user can see the folder.

This does not mean the user can actually access its contents. It just determines if it is visible to him or not.

**category**

**category\_id**

**contribution**

**contribution\_id**

**description**

The description of the folder

**event**

**event\_id**

**events\_backref\_name** = 'all\_attachment\_folders'

**classmethod get\_for\_linked\_object** (*linked\_object*, *preload\_event=False*)

Get the attachments for the given object.

This only returns attachments that haven't been deleted.

#### Parameters

- **linked\_object** – A category, event, session, contribution or subcontribution.
- **preload\_event** – If all attachments for the same event should be pre-loaded and cached in the app context. This must not be used when **linked\_object** is a category.

**classmethod get\_or\_create** (*linked\_object*, *title=None*)

Get a folder for the given object or create it.

If no folder title is specified, the default folder will be used. It is the caller's responsibility to add the folder or an object (such as an attachment) associated with it to the SQLAlchemy session using `db.session.add(...)`.

**classmethod get\_or\_create\_default** (*linked\_object*)

Get the default folder for the given object or creates it.

**id**

The ID of the folder

**is\_always\_visible**

If the folder is always visible (even if you cannot access it)

**is\_default**

If the folder is the default folder (used for “folder-less” files)

**is\_deleted**

If the folder has been deleted



**is\_hidden**

If the folder is never shown in the frontend (even if you can access it)

**link\_backref\_lazy** = 'dynamic'

**link\_backref\_name** = 'attachment\_folders'

**link\_type**

**linked\_event**

**linked\_event\_id**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**own\_no\_access\_contact** = None

**protection\_mode**

**protection\_parent**

The parent object to consult for `ProtectionMode.inheriting`.

**session**

**session\_block** = None

**session\_block\_id** = None

**session\_id**

**subcontribution**

**subcontribution\_id**

**title**

The name of the folder (None for the default folder)

**unique\_links** = 'is\_default'

```
class indico.modules.attachments.models.principals.AttachmentFolderPrincipal(**kwargs)
    Bases: indico.core.db.sqlalchemy.principals.PrincipalMixin, sqlalchemy.orm.
    decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_category_roles = True
allow_event_roles = True
allow_registration_forms = True
category_role
category_role_id
email = None
event_role
event_role_id
folder_id
    The ID of the associated folder
id
    The ID of the acl entry
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = 'in_attachment_folder_acls'
registration_form
registration_form_id
type
unique_columns = ('folder_id',)
user
user_id
```

```
class indico.modules.attachments.models.principals.AttachmentPrincipal(**kwargs)
Bases: indico.core.db.sqlalchemy.principals.PrincipalMixin, sqlalchemy.orm.
decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_category_roles = True
allow_event_roles = True
allow_registration_forms = True
```

```

attachment_id
    The ID of the associated attachment
category_role
category_role_id
email = None
event_role
event_role_id
id
    The ID of the acl entry
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = 'in_attachment_acls'
registration_form
registration_form_id
type
unique_columns = ('attachment_id',)
user
user_id

```

## Operations

`indico.modules.attachments.operations.add_attachment_link` (*data*, *linked\_object*)  
 Add a link attachment to *linked\_object*.

## Utilities

`indico.modules.attachments.util.can_manage_attachments` (*obj*, *user*, *allow\_admin=True*)  
 Check if a user can manage attachments for the object.

`indico.modules.attachments.util.get_attached_folders` (*linked\_object*, *include\_empty=True*, *include\_hidden=True*, *preload\_event=False*)  
 Return a list of all the folders linked to an object.

### Parameters

- **linked\_object** – The object whose attachments are to be returned
- **include\_empty** – Whether to return empty folders as well.

- **include\_hidden** – Include folders that the user can't see
- **preload\_event** – in the process, preload all objects tied to the corresponding event and keep them in cache

```
indico.modules.attachments.util.get_attached_items(linked_object,           in-
                                                    clude_empty=True,         in-
                                                    clude_hidden=True,
                                                    preload_event=False)
```

Return a structured representation of all the attachments linked to an object.

#### Parameters

- **linked\_object** – The object whose attachments are to be returned
- **include\_empty** – Whether to return empty folders as well.
- **include\_hidden** – Include folders that the user can't see
- **preload\_event** – in the process, preload all objects tied to the corresponding event and keep them in cache

```
indico.modules.attachments.util.get_default_folder_names()
```

```
indico.modules.attachments.util.get_event(linked_object)
```

```
class indico.modules.attachments.preview.ImagePreviewer
```

Bases: *indico.modules.attachments.preview.Previewer*

```
ALLOWED_CONTENT_TYPE = re.compile('^image/')
```

```
TEMPLATE = 'image_preview.html'
```

```
class indico.modules.attachments.preview.MarkdownPreviewer
```

Bases: *indico.modules.attachments.preview.Previewer*

```
ALLOWED_CONTENT_TYPE = re.compile('^text/markdown$')
```

```
classmethod generate_content(attachment)
```

Generate the HTML output of the file preview.

```
class indico.modules.attachments.preview.PDFPreviewer
```

Bases: *indico.modules.attachments.preview.Previewer*

```
ALLOWED_CONTENT_TYPE = re.compile('^application/pdf$')
```

```
TEMPLATE = 'iframe_preview.html'
```

```
classmethod can_preview(attachment_file)
```

Check if the content type of the file matches the allowed content type of files that the previewer can be used for.

```
class indico.modules.attachments.preview.Previewer
```

Bases: *object*

Base class for file previewers.

To create a new file previewer, subclass this class and register it using the *get\_file\_previewers* signal.

```
ALLOWED_CONTENT_TYPE = None
```

```
TEMPATE = None
```

```
TEMPLATES_DIR = 'attachments/previewers/'
```

```

classmethod can_preview (attachment_file)
    Check if the content type of the file matches the allowed content type of files that the previewer can be
    used for.

classmethod generate_content (attachment)
    Generate the HTML output of the file preview.

class indico.modules.attachments.preview.TextPreviewer
    Bases: indico.modules.attachments.preview.Previewer

    ALLOWED_CONTENT_TYPE = re.compile('^text/plain$')

    classmethod generate_content (attachment)
        Generate the HTML output of the file preview.

indico.modules.attachments.preview.get_file_previewer (attachment_file)
    Return a file previewer for the given attachment file based on the file's content type.

indico.modules.attachments.preview.get_file_previewers ()

```

## 7.1.24 Room booking

---

**Todo:** Docstrings (module, models, utilities, services)

---

### Models

```

class indico.modules.rb.models.rooms.Room (**kwargs)
    Bases: indico.core.db.sqlalchemy.protection.ProtectionManagersMixin,
            sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    access_key = None

    acl_entries

    attributes

    available_equipment

    blocked_rooms

    bookable_hours

    booking_limit_days

    building

    can_access (user, allow_admin=True)
        Check if the user can access the object.

        Parameters
        • user – The User to check. May be None if the user is not logged in.
        • allow_admin – If admin users should always have access

```

**can\_book** (*user*, *allow\_admin=True*)

**can\_delete** (*user*)

**can\_edit** (*user*)

**can\_manage** (*user*, *permission=None*, *allow\_admin=True*, *check\_parent=True*, *explicit\_permission=False*)

Check if the user can manage the object.

#### Parameters

- **user** – The *User* to check. May be *None* if the user is not logged in.
- **allow\_admin** – If admin users should always have access
- **check\_parent** – If the parent object should be checked. In this case the permission is ignored; only full management access is inherited to children.
- **explicit\_permission** – If the specified permission should be checked explicitly instead of short-circuiting the check for Indico admins or managers. When this option is set to *True*, the values of *allow\_admin* and *check\_parent* are ignored. This also applies if *permission* is *None* in which case this argument being set to *True* is equivalent to *allow\_admin* and *check\_parent* being set to *False*.

**Param permission:** The management permission that is needed for the check to succeed. If not specified, full management privs are required. May be set to the string 'ANY' to check if the user has any management privileges. If the user has *full\_access* privileges, he's assumed to have all possible permissions.

**can\_moderate** (*user*, *allow\_admin=True*)

**can\_override** (*user*, *allow\_admin=True*)

**can\_prebook** (*user*, *allow\_admin=True*)

**capacity**

**check\_advance\_days** (*end\_date*, *user=None*, *quiet=False*)

**check\_bookable\_hours** (*start\_time*, *end\_time*, *user=None*, *quiet=False*)

**comments**

**default\_protection\_mode** = 0

**details\_url**

**disallowed\_protection\_modes** = frozenset({<ProtectionMode.inheriting: 1>})

**division**

**end\_notification\_daily**

**end\_notification\_monthly**

**end\_notification\_weekly**

**end\_notifications\_enabled**

**favorite\_of**

**static filter\_available** (*start\_dt*, *end\_dt*, *repetition*, *include\_blockings=True*, *include\_pre\_bookings=True*, *include\_pending\_blockings=False*)

Return a SQLAlchemy filter criterion ensuring that the room is available during the given time.

**static filter\_bookable\_hours** (*start\_time*, *end\_time*)

```

static filter_nonbookable_periods (start_dt, end_dt)
classmethod find_with_attribute (attribute)
    Search rooms which have a specific attribute.
floor
full_name
generate_name ()
get_attribute_by_name (attribute_name)
get_attribute_value (name, default=None)
get_blocked_rooms (*dates, **kwargs)
classmethod get_permissions_for_user (user, allow_admin=True)
    Get the permissions for all rooms for a user.

    In case of multipass-based groups it will try to get a list of all groups the user is in, and if that's not possible
    check the permissions one by one for each room (which may result in many group membership lookups).

    It is recommended to not call this in any place where performance matters and to memoize the result.
static get_with_data (*args, **kwargs)
has_attribute (attribute_name)
has_equipment (*names)
has_photo
id
is_auto_confirm
is_deleted
is_reservable
static is_user_admin (user)
key_location
latitude
location
location_id
location_name
longitude
map_url
max_advance_days
name
nonbookable_periods
notification_before_days
notification_before_days_monthly
notification_before_days_weekly
notification_emails

```

**notifications\_enabled**

**number**

**own\_no\_access\_contact** = None

**owner**

The owner of the room. This is purely informational and does not grant any permissions on the room.

**owner\_id**

**photo**

**photo\_id**

**protection\_mode**

**protection\_parent**

The parent object to consult for ProtectionMode.inheriting.

**reservations**

**reservations\_need\_confirmation**

**set\_attribute\_value** (*name*, *value*)

**site**

**sprite\_position**

**surface\_area**

**telephone**

**verbose\_name**

Verbose name for the room (long)

**class** indico.modules.rb.models.room\_attributes.**RoomAttribute** (\*\**kwargs*)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

**is\_hidden**

**name**

**title**

**class** indico.modules.rb.models.room\_attributes.**RoomAttributeAssociation** (\*\**kwargs*)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**attribute**

**attribute\_id**



**room\_id****value****class** indico.modules.rb.models.room\_bookable\_hours.**BookableHours** (\*\*kwargs)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**end\_time****fits\_period** (st, et)**room\_id****start\_time****class** indico.modules.rb.models.room\_nonbookable\_periods.**NonBookablePeriod** (\*\*kwargs)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**end\_dt****overlaps** (st, et)**room\_id****start\_dt****class** indico.modules.rb.models.blockings.**Blocking** (\*\*kwargs)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**allowed**

A descriptor that presents a read/write view of an object attribute.

**blocked\_rooms****can\_delete** (user, allow\_admin=True)**can\_edit** (user, allow\_admin=True)**can\_override** (user, room=None, explicit\_only=False, allow\_admin=True)

Check if a user can override the blocking.

The following persons are authorized to override a blocking: - the creator of the blocking - anyone on the blocking's ACL - unless explicit\_only is set: rb admins and room managers (if a room is given)

**created\_by\_id**

**created\_by\_user**

The user who created this blocking.

**created\_dt**

**end\_date**

**external\_details\_url**

**id**

**is\_active\_at** (*d*)

**reason**

**start\_date**

**class** indico.modules.rb.models.blocked\_rooms.**BlockedRoom** (*\*\*kwargs*)

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**State**

alias of *BlockedRoomState*

**approve** (*notify\_blocker=True*)

Approve the room blocking, rejecting all colliding reservations/occurrences.

**blocking\_id**

**id**

**reject** (*user=None, reason=None*)

Reject the room blocking.

**rejected\_by**

**rejection\_reason**

**room\_id**

**state**

**state\_name**

**class** indico.modules.rb.models.blocked\_rooms.**BlockedRoomState**

Bases: indico.util.enum.RichIntEnum

An enumeration.

**accepted** = 1

**pending** = 0

**rejected** = 2

**class** indico.modules.rb.models.blocking\_principals.**BlockingPrincipal** (*\*\*kwargs*)

Bases: indico.core.db.sqlalchemy.principals.PrincipalMixin, sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
blocking_id
category_role = None
category_role_id = None
email = None
event_role = None
event_role_id = None
id
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = 'in_blocking_acls'
registration_form = None
registration_form_id = None
type
unique_columns = ('blocking_id',)
user
user_id
```

```
class indico.modules.rb.models.equipment.EquipmentType (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
features
id
name
```

```
class indico.modules.rb.models.locations.Location (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
id
```

**is\_deleted**

**map\_url\_template**

**name**

**room\_name\_format**

Translate Postgres' format syntax (e.g. `%1$s/%2$s-%3$s`) to Python's.

**rooms**

**class** `indico.modules.rb.models.map_areas.MapArea` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**bottom\_right\_latitude**

**bottom\_right\_longitude**

**id**

**is\_default**

**name**

**top\_left\_latitude**

**top\_left\_longitude**

**class** `indico.modules.rb.models.photos.Photo` (*\*\*kwargs*)

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**data**

**id**

**exception** `indico.modules.rb.models.reservations.ConflictingOccurrences`

Bases: `Exception`

**class** `indico.modules.rb.models.reservations.RepeatFrequency`

Bases: `int`, `indico.util.enum.IndicoEnum`

An enumeration.

**DAY = 1**

**MONTH = 3**

**NEVER = 0**

**WEEK = 2**

**class** `indico.modules.rb.models.reservations.RepeatMapping`

Bases: `object`

**classmethod** `get_message` (*repeat\_frequency*, *repeat\_interval*)

```

    classmethod get_short_name (repeat_frequency, repeat_interval)

    mapping = {(<RepeatFrequency.NEVER: 0>, 0): ('Single reservation', None, 'none'), (<R
class indico.modules.rb.models.reservations.Reservation (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

    accept (user, reason=None)

    add_edit_log (edit_log)

    booked_for_id

    booked_for_name

    booked_for_user
        The user this booking was made for. Assigning a user here also updates booked_for_name.

    booking_reason

    can_accept (user, allow_admin=True)

    can_cancel (user, allow_admin=True)

    can_delete (user, allow_admin=True)

    can_edit (user, allow_admin=True)

    can_reject (user, allow_admin=True)

    cancel (user, reason=None, silent=False)

    contact_email

    classmethod create_from_data (room, data, user, prebook=None, ignore_admin=False)
        Create a new reservation.

        Parameters
        • room – The Room that's being booked.
        • data – A dict containing the booking data, usually from a NewBookingConfirmForm
          instance
        • user – The User who creates the booking.
        • prebook – Instead of determining the booking type from the user's permissions, always
          use the given mode.
        • ignore_admin – Whether to ignore the user's admin status.

    create_occurrences (skip_conflicts, user=None, allow_admin=True)

    created_by_id

    created_by_user
        The user who created this booking.

    created_dt

    edit_logs

```

`end_dt`  
`end_notification_sent`  
`event`  
`external_details_url`  
`find_excluded_days()`  
`find_overlapping()`  
`static find_overlapping_with(room, occurrences, skip_reservation_id=None)`  
`get_conflicting_occurrences()`  
`static get_with_data(*args, **kwargs)`  
`id`  
`is_accepted`  
`is_archived`  
`is_booked_for(user)`  
`is_cancelled`  
`is_owned_by(user)`  
`is_pending`  
`is_rejected`  
`is_repeating`  
`link`  
`link_id`  
`linked_object`  
`location_name`  
`modify(data, user)`  
    Modify an existing reservation.

**Parameters**

- **data** – A dict containing the booking data, usually from a `ModifyBookingForm` instance
- **user** – The *User* who modifies the booking.

`occurrences`  
`reject(user, reason, silent=False)`  
`rejection_reason`  
`repeat_frequency`  
`repeat_interval`  
`repetition`  
`reset_approval(user)`  
`room_id`  
`start_dt`

**state**

```
class indico.modules.rb.models.reservations.ReservationLink (**kwargs)
    Bases: indico.core.db.sqlalchemy.links.LinkMixin, sqlalchemy.orm.decl_api.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allowed_link_types = {<LinkType.event: 2>, <LinkType.contribution: 3>, <LinkType.session: 4>}
```

```
category = None
```

```
category_id = None
```

```
contribution
```

```
contribution_id
```

```
event
```

```
event_id
```

```
events_backref_name = 'all_room_reservation_links'
```

```
id
```

```
link_backref_name = 'room_reservation_links'
```

```
link_type
```

```
linked_event
```

```
linked_event_id
```

```
session = None
```

```
session_block
```

```
session_block_id
```

```
session_id = None
```

```
subcontribution = None
```

```
subcontribution_id = None
```

```
class indico.modules.rb.models.reservations.ReservationState
```

```
    Bases: int, indico.util.enum.IndicoEnum
```

An enumeration.

```
accepted = 2
```

```
cancelled = 3
```

```
pending = 1
```

```
rejected = 4
```

```
class indico.modules.rb.models.reservation_edit_logs.ReservationEditLog (**kwargs)
```

```
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

`id`  
`info`  
`reservation_id`  
`timestamp`  
`user_name`

```
class indico.modules.rb.models.reservation_occurrences.ReservationOccurrence (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
NO_RESERVATION_USER_STRATEGY = <sqlalchemy.orm.strategy_options._UnboundLoad object>
```

A relationship loading strategy that will avoid loading the users linked to a reservation. You want to use this in pretty much all cases where you eager-load the *reservation* relationship.

```
can_cancel (user, allow_admin=True)
can_reject (user, allow_admin=True)
cancel (user, reason=None, silent=False)
classmethod create_series (start, end, repetition)
classmethod create_series_for_reservation (reservation)
date
end_dt
external_cancellation_url
static filter_overlap (occurrences)
classmethod find_overlapping_with (room, occurrences, skip_reservation_id=None)
get_overlap (occurrence, skip_self=False)
is_cancelled
is_rejected
is_valid
is_within_cancel_grace_period
classmethod iter_create_occurrences (start, end, repetition)
static iter_start_time (start, end, repetition)
notification_sent
overlaps (occurrence, skip_self=False)
reject (user, reason, silent=False)
rejection_reason
reservation_id
```



**start\_dt**

**state**

**class** `indico.modules.rb.models.reservation_occurrences.ReservationOccurrenceState`  
 Bases: `int`, `indico.util.enum.IndicoEnum`

An enumeration.

**cancelled = 3**

**rejected = 4**

**valid = 2**

`indico.modules.rb.models.util.proxy_to_reservation_if_last_valid_occurrence(f)`  
 Forward a method call to `self.reservation` if there is only one occurrence.

## Utilities

`indico.modules.rb.util.TempReservationConcurrentOccurrence`  
 alias of `indico.modules.rb.util.ReservationOccurrenceTmp`

`indico.modules.rb.util.TempReservationOccurrence`  
 alias of `indico.modules.rb.util.ReservationOccurrenceTmp`

`indico.modules.rb.util.build_rooms_spritesheet()`

`indico.modules.rb.util.generate_spreadsheet_from_occurrences(occurrences)`  
 Generate spreadsheet data from a given booking occurrence list.

**Parameters** `occurrences` – The booking occurrences to include in the spreadsheet

`indico.modules.rb.util.get_booking_params_for_event(event)`  
 Get a set of RB interface parameters suitable for this event.

These parameters can then be used to construct a URL that will lead to a pre-filled search that matches the start/end times for a given day.

**Parameters** `event` – *Event* object

`indico.modules.rb.util.get_linked_object(type_, id_)`

`indico.modules.rb.util.get_prebooking_collisions(reservation)`

`indico.modules.rb.util.get_resized_room_photo(room)`

`indico.modules.rb.util.group_by_occurrence_date(occurrences, sort_by=None)`

`indico.modules.rb.util.is_booking_start_within_grace_period(start_dt, user, allow_admin=False)`

`indico.modules.rb.util.rb_check_user_access(user)`  
 Check if the user has access to the room booking system.

`indico.modules.rb.util.rb_is_admin(user)`  
 Check if the user is a room booking admin.

`indico.modules.rb.util.remove_room_spritesheet_photo(room)`

`indico.modules.rb.util.serialize_availability(availability)`

`indico.modules.rb.util.serialize_blockings(data)`

`indico.modules.rb.util.serialize_booking_details(booking)`

```
indico.modules.rb.util.serialize_concurrent_pre_bookings(data)
indico.modules.rb.util.serialize_nonbookable_periods(data)
indico.modules.rb.util.serialize_occurrences(data)
indico.modules.rb.util.serialize_unbookable_hours(data)
indico.modules.rb.statistics.calculate_rooms_bookable_time(rooms,
                                                         start_date=None,
                                                         end_date=None)
indico.modules.rb.statistics.calculate_rooms_booked_time(rooms, start_date=None,
                                                         end_date=None)
indico.modules.rb.statistics.calculate_rooms_occupancy(rooms, start=None,
                                                         end=None)
```

## 7.1.25 Authentication

---

**Todo:** Docstrings (module, models, utilities)

---

### Models

**class** `indico.modules.auth.models.identities.Identity` (\*\*kwargs)

Bases: `sqlalchemy.orm.decl_api.Model`

Identities of Indico users.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**data**

**id**

the unique id of the identity

**identifier**

the unique identifier of the user within its provider

**last\_login\_dt**

the timestamp of the latest login

**last\_login\_ip**

the ip address that was used for the latest login

**locator**

**multipass\_data**

internal data used by the flask-multipass system

**password**

the password of the user in case of a local identity

**password\_hash**

the hash of the password in case of a local identity

**provider**  
the provider name of the identity

**register\_login** (*ip*)  
Update the last login information.

**safe\_last\_login\_dt**  
last\_login\_dt that is safe for sorting (no None values).

**user\_id**  
the id of the user this identity belongs to

**class** indico.modules.auth.models.registration\_requests.**RegistrationRequest** (*\*\*kwargs*)  
Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**comment**

**email**

**extra\_emails**

**id**

**identity\_data**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**settings**

**user\_data**

## Utilities

indico.modules.auth.util.**impersonate\_user** (*user*)  
Impersonate another user as an admin.

`indico.modules.auth.util.load_identity_info()`

Retrieve identity information from the session.

`indico.modules.auth.util.redirect_to_login(next_url=None, reason=None)`

Redirect to the login page.

#### Parameters

- **next\_url** – URL to be redirected upon successful login. If not specified, it will be set to `request.relative_url`.
- **reason** – Why the user is redirected to a login page.

`indico.modules.auth.util.register_user(email, extra_emails, user_data, identity_data, settings, from_moderation=False)`

Create a user based on the registration data provided during the user registration process (via *RHRegister* and *RegistrationHandler*).

This method is not meant to be used for generic user creation, the only reason why this is here is that approving a registration request is handled by the *users* module.

`indico.modules.auth.util.save_identity_info(identity_info, user)`

Save information from IdentityInfo in the session.

`indico.modules.auth.util.undo_impersonate_user()`

Undo an admin impersonation login and revert to the old user.

`indico.modules.auth.util.url_for_login(next_url=None)`

`indico.modules.auth.util.url_for_logout(next_url=None)`

`indico.modules.auth.util.url_for_register(next_url=None, email=None)`

Returns the URL to register

#### Parameters

- **next\_url** – The URL to redirect to afterwards.
- **email** – A pre-validated email address to use when creating a new local account. Use this argument **ONLY** when sending the link in an email or if the email address has already been validated using some other way.

## 7.1.26 OAuth

---

**Todo:** Docstrings (module, models, provider)

---

### Models

**class** `indico.core.oauth.models.applications.OAuthApplication(**kwargs)`

Bases: `authlib.oauth2.rfc6749.models.ClientMixin`, `sqlalchemy.orm.decl_api.Model`

OAuth applications registered in Indico.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**allow\_pkce\_flow**

whether the application can use the PKCE flow without a client secret

**allowed\_scopes**

the OAuth scopes the application may request access to

**check\_client\_secret** (*client\_secret*)

Check *client\_secret* matching with the client. For instance, in the client table, the column is called *client\_secret*:

```
def check_client_secret(self, client_secret):
    return self.client_secret == client_secret
```

**Parameters** *client\_secret* – A string of client secret

**Returns** bool

**check\_endpoint\_auth\_method** (*method*, *endpoint*)

Check if client support the given method for the given endpoint. There is a *token\_endpoint\_auth\_method* defined via [RFC7591](#). Developers MAY re-implement this method with:

```
def check_endpoint_auth_method(self, method, endpoint):
    if endpoint == 'token':
        # if client table has `token_endpoint_auth_method`
        return self.token_endpoint_auth_method == method
    return True
```

Method values defined by this specification are:

- **“none”**: The client is a public client as defined in OAuth 2.0, and does not have a client secret.
- **“client\_secret\_post”**: The client uses the HTTP POST parameters as defined in OAuth 2.0
- **“client\_secret\_basic”**: The client uses HTTP Basic as defined in OAuth 2.0

**check\_grant\_type** (*grant\_type*)

Validate if the client can handle the given *grant\_type*. There are four grant types defined by RFC6749:

- *authorization\_code*
- *implicit*
- *client\_credentials*
- *password*

For instance, there is a *allowed\_grant\_types* column in your client:

```
def check_grant_type(self, grant_type):
    return grant_type in self.grant_types
```

**Parameters** *grant\_type* – the requested *grant\_type* string.

**Returns** bool

**check\_redirect\_uri** (*redirect\_uri*)

Called by authlib to validate the *redirect\_uri*.

Uses a logic similar to the one at GitHub, i.e. protocol and host/port must match exactly and if there is a path in the whitelisted URL, the path of the *redirect\_uri* must start with that path.

**check\_response\_type** (*response\_type*)

Validate if the client can handle the given response\_type. There are two response types defined by RFC6749: code and token. For instance, there is a `allowed_response_types` column in your client:

```
def check_response_type(self, response_type):
    return response_type in self.response_types
```

**Parameters** `response_type` – the requested response\_type string.

**Returns** bool

**client\_id**

the OAuth client\_id

**client\_secret**

the OAuth client\_secret

**default\_redirect\_uri****description**

human readable description

**get\_allowed\_scope** (*scope*)

A method to return a list of requested scopes which are supported by this client. For instance, there is a `scope` column:

```
def get_allowed_scope(self, scope):
    if not scope:
        return ''
    allowed = set(scope_to_list(self.scope))
    return list_to_scope([s for s in scope.split() if s in allowed])
```

**Parameters** `scope` – the requested scope.

**Returns** string of scope

**get\_client\_id()**

A method to return client\_id of the client. For instance, the value in database is saved in a column called `client_id`:

```
def get_client_id(self):
    return self.client_id
```

**Returns** string

**get\_default\_redirect\_uri()**

A method to get client default redirect\_uri. For instance, the database table for client has a column called `default_redirect_uri`:

```
def get_default_redirect_uri(self):
    return self.default_redirect_uri
```

**Returns** A URL string

**id**

the unique id of the application

**is\_enabled**  
whether the application is enabled or disabled

**is\_trusted**  
whether the application can access user data without asking for permission

**locator**

**name**  
human readable name

**redirect\_uris**  
the OAuth absolute URIs that a application may use to redirect to after authorization

**reset\_client\_secret()**

**system\_app\_type**  
the type of system app (if any). system apps cannot be deleted

```
class indico.core.oauth.models.applications.OAuthApplicationUserLink (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model
```

The authorization link between an OAuth app and a user.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**application**

**application\_id**

**id**

**scopes**

**update\_scopes** (*scopes: set*)

**user**

**user\_id**

```
class indico.core.oauth.models.applications.SystemAppType
    Bases: int, indico.util.enum.IndicoEnum
```

An enumeration.

**checkin** = 1

**default\_data**

**enforced\_data**

**none** = 0

```
class indico.core.oauth.models.tokens.OAuth2AuthorizationCode (code:          str,
                                                                user_id:       int,
                                                                client_id:    str,
                                                                code_challenge:
                                                                str,
                                                                code_challenge_method:
                                                                str, redirect_uri:
                                                                str = "", scope: str
                                                                = "", auth_time:
                                                                datetime.datetime
                                                                = <factory>)
```

Bases: `authlib.oauth2.rfc6749.models.AuthorizationCodeMixin`

**get\_auth\_time()**

**get\_nonce()**

**get\_redirect\_uri()**

A method to get authorization code's `redirect_uri`. For instance, the database table for authorization code has a column called `redirect_uri`:

```
def get_redirect_uri(self):
    return self.redirect_uri
```

**Returns** A URL string

**get\_scope()**

A method to get scope of the authorization code. For instance, the column is called `scope`:

```
def get_scope(self):
    return self.scope
```

**Returns** scope string

**is\_expired()**

**redirect\_uri** = ''

**scope** = ''

```
class indico.core.oauth.models.tokens.OAuthToken (**kwargs)
```

Bases: `indico.core.oauth.models.tokens.TokenModelBase`

OAuth tokens.

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**access\_token\_hash**

**app\_user\_link**

**app\_user\_link\_id**

**application**



**check\_client** (*client*)

A method to check if this token is issued to the given client. For instance, `client_id` is saved on token table:

```
def check_client(self, client):
    return self.client_id == client.client_id
```

Returns bool

**created\_dt****get\_scope** ()

A method to get scope of the authorization code. For instance, the column is called `scope`:

```
def get_scope(self):
    return self.scope
```

Returns scope string

**id****is\_revoked** ()

A method to define if this token is revoked. For instance, there is a boolean column `revoked` in the table:

```
def is_revoked(self):
    return self.revoked
```

Returns boolean

**last\_used\_dt****last\_used\_ip****use\_count****user**

**class** `indico.core.oauth.models.tokens.TokenModelBase` (*\*\*kwargs*)

Bases: `authlib.oauth2.rfc6749.models.TokenMixin`, `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**access\_token**

Similar to *PasswordProperty* but tailored towards API tokens.

Since tokens are used much more often than passwords, they use a fast hash algorithm instead of a secure one. This is not a problem for tokens as they are fully random and much longer than the typical password or even passphrase.

**access\_token\_hash** = `Column(None, String(), table=None, nullable=False)`

**created\_dt** = `Column(None, UTCDateTime(), table=None, nullable=False, default=ColumnDef`

**get\_expires\_in** ()

A method to get the `expires_in` value of the token. e.g. the column is called `expires_in`:

```
def get_expires_in(self):  
    return self.expires_in
```

**Returns** timestamp int

`id = Column(None, Integer(), table=None, primary_key=True, nullable=False)`

`is_expired()`

A method to define if this token is expired. For instance, there is a column `expired_at` in the table:

```
def is_expired(self):  
    return self.expired_at < now
```

**Returns** boolean

`last_used_dt = Column(None, UTCDateTime(), table=None)`

`last_used_ip = Column(None, INET(), table=None)`

`locator`

**scopes**

The set of scopes this token has access to.

`use_count = Column(None, Integer(), table=None, nullable=False, default=ColumnDefault(`

## 7.1.27 Group

---

**Todo:** Docstrings (module)

---

### Models

**class** `indico.modules.groups.models.groups.LocalGroup` (*\*\*kwargs*)  
Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**id**

the unique id of the group

**members**

the users in the group

**name**

the name of the group

**proxy**

Return a GroupProxy wrapping this group.

**class** `indico.modules.groups.core.GroupProxy`

Bases: `object`

Provide a generic interface for both local and multipass groups.

Creating an instance of this class actually creates either a `LocalGroupProxy` or a `MultipassGroupProxy`, but they expose the same API.

#### Parameters

- **name\_or\_id** – The name of a multipass group or ID of a local group
- **provider** – The provider of a multipass group

Create the correct `GroupProxy` for the group type.

**as\_principal**

The serializable principal identifier of this group.

**get\_members** ()

Get the list of users who are members of the group.

**group**

The underlying group object.

**has\_member** (*user*)

Check if the user is a member of the group.

This can also be accessed using the `in` operator.

**identifier**

**principal\_order** = 3

**classmethod** **search** (*name*, *exact=False*, *providers=None*)

Search for groups.

#### Parameters

- **name** – The group name to search for.
- **exact** – If only exact matches should be found (much faster)
- **providers** – `None` to search in all providers and local groups. May be a set specifying providers to search in. For local groups, the 'indico' provider name may be used.

## Utilities

`indico.modules.groups.util.serialize_group` (*group*)

Serialize group to JSON-like object.

## 7.1.28 Video conference

---

**Todo:** Docstrings (module, models, utilities, plugins, exceptions)

---

## Models

```
class indico.modules.vc.models.vc_rooms.VCRoom (**kwargs)
```

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**created\_by\_id**

ID of the creator

**created\_by\_user**

The user who created the videoconference room

**created\_dt**

Creation timestamp of the videoconference room

**data**

videoconference plugin-specific data

**id**

Videoconference room ID

**locator**

**modified\_dt**

Modification timestamp of the videoconference room

**name**

Name of the videoconference room

**plugin**

**status**

Status of the videoconference room

**type**

Type of the videoconference room

```
class indico.modules.vc.models.vc_rooms.VCRoomEventAssociation (**kwargs)
```

Bases: sqlalchemy.orm.decl\_api.Model

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**contribution\_id**

**data**

videoconference plugin-specific data

**delete** (*user*, *delete\_all=False*)

Delete a VC room from an event.

If the room is not used anywhere else, the room itself is also deleted.

### Parameters

- **user** – the user performing the deletion

- **delete\_all** – if True, the room is detached from all events and deleted.

**event**

The associated Event

**event\_id**

ID of the event

**classmethod find\_for\_event** (*event*, *include\_hidden=False*, *include\_deleted=False*,  
*only\_linked\_to\_event=False*, *\*\*kwargs*)

Return a Query that retrieves the videoconference rooms for an event.

**Parameters**

- **event** – an indico Event
- **only\_linked\_to\_event** – only retrieve the vc rooms linked to the whole event
- **kwargs** – extra kwargs to pass to `filter_by()`

**classmethod get\_linked\_for\_event** (*event*)

Get a dict mapping link objects to event vc rooms.

**id**

Association ID

**link\_object****link\_type**

Type of the object the vc\_room is linked to

**linked\_block**

The linked session block (if the VC room is attached to a block)

**linked\_contrib**

The linked contribution (if the VC room is attached to a contribution)

**linked\_event**

The linked event (if the VC room is attached to the event itself)

**linked\_event\_id****locator**

**classmethod register\_link\_events** ()

**session\_block\_id****show**

If the vc room should be shown on the event page

**vc\_room**

The associated :class:VCRoom

**vc\_room\_id**

ID of the videoconference room

**class** `indico.modules.vc.models.vc_rooms.VCRoomLinkType`

Bases: `int`, `indico.util.enum.IndicoEnum`

An enumeration.

**block** = 3

**contribution** = 2

**event** = 1

```
class indico.modules.vc.models.vc_rooms.VCRoomStatus
    Bases: int, indico.util.enum.IndicoEnum

    An enumeration.

    created = 1
    deleted = 2
```

## Utilities

`indico.modules.vc.util.find_event_vc_rooms` (*from\_dt=None, to\_dt=None, distinct=False*)  
Find VC rooms matching certain criteria.

### Parameters

- **from\_dt** – earliest event/contribution to include
- **to\_dt** – latest event/contribution to include
- **distinct** – if True, never return the same (*event*, *vcroom*) more than once (even if it's linked more than once to that event)

`indico.modules.vc.util.get_linked_to_description` (*obj*)

`indico.modules.vc.util.get_managed_vc_plugins` (*user*)  
Return the plugins the user can manage.

`indico.modules.vc.util.get_vc_plugins` ()  
Return a dict containing the available videoconference plugins.

`indico.modules.vc.util.resolve_title` (*obj*)

## Plugins

```
class indico.modules.vc.plugins.VCPluginMixin
    Bases: object

    acl_settings = {'acl', 'managers'}

    can_manage_vc (user)
        Check if a user has management rights on this VC system.

    can_manage_vc_room (user, room)
        Check if a user can manage a vc room.

    can_manage_vc_rooms (user, event)
        Check if a user can manage vc rooms on an event.

    category = 'Videoconference'

    clone_room (old_event_vc_room, link_object)
        Clone the room, returning a new VCRoomEventAssociation.
```

### Parameters

- **old\_event\_vc\_room** – the original VCRoomEventAssociation
- **link\_object** – the new object the association will be tied to

**Returns** the new VCRoomEventAssociation

**create\_form** (*event, existing\_vc\_room=None, existing\_event\_vc\_room=None*)  
Create the videoconference room form.

**Parameters**

- **event** – the event the videoconference room is for
- **existing\_vc\_room** – a `vc_room` from which to retrieve data for the form

**Returns** an instance of an `IndicoForm` subclass

**create\_room** (*vc\_room, event*)

**default\_settings** = {'notification\_emails': []}

**friendly\_name** = None

the readable name of the VC plugin

**get\_extra\_delete\_msg** (*vc\_room, event\_vc\_room*)

Return a custom message to show in the confirmation dialog when deleting a VC room.

**Parameters**

- **vc\_room** – the VC room object
- **event\_vc\_room** – the association of an event and a VC room

**Returns** a string (may contain HTML) with the message to display

**get\_notification\_bcc\_list** (*action, vc\_room, event*)

**get\_notification\_cc\_list** (*action, vc\_room, event*)

**get\_vc\_room\_attach\_form\_defaults** (*event*)

**get\_vc\_room\_form\_defaults** (*event*)

**icon\_url**

**init** ()

**logo\_url**

**render\_buttons** (*vc\_room, event\_vc\_room, \*\*kwargs*)

Render a list of plugin specific buttons (eg: Join URL, etc) in the management area.

**Parameters**

- **vc\_room** – the VC room object
- **event\_vc\_room** – the association of an event and a VC room
- **kwargs** – arguments passed to the template

**render\_event\_buttons** (*vc\_room, event\_vc\_room, \*\*kwargs*)

Render a list of plugin specific buttons (eg: Join URL, etc) in the event page.

**Parameters**

- **vc\_room** – the VC room object
- **event\_vc\_room** – the association of an event and a VC room
- **kwargs** – arguments passed to the template

**render\_form** (*\*\*kwargs*)

Render the videoconference room form.

**Parameters** **kwargs** – arguments passed to the template

**render\_info\_box** (*vc\_room, event\_vc\_room, event, \*\*kwargs*)

Render the information shown in the expandable box of a VC room row.

**Parameters**

- **vc\_room** – the VC room object
- **event\_vc\_room** – the association of an event and a VC room
- **event** – the event with the current VC room attached to it
- **kwargs** – arguments passed to the template

**render\_manage\_event\_info\_box** (*vc\_room, event\_vc\_room, event, \*\*kwargs*)

Render the information shown in the expandable box on a VC room in the management area.

**Parameters**

- **vc\_room** – the VC room object
- **event\_vc\_room** – the association of an event and a VC room
- **event** – the event with the current VC room attached to it
- **kwargs** – arguments passed to the template

**service\_name**

**settings\_form**

alias of `indico.modules.vc.forms.VCPluginSettingsFormBase`

**update\_data\_association** (*event, vc\_room, event\_vc\_room, data*)

**update\_data\_vc\_room** (*vc\_room, data, is\_new=False*)

**vc\_room\_attach\_form** = `None`

the `IndicoForm` to use for the videoconference room attach form

**vc\_room\_form** = `None`

the `IndicoForm` to use for the videoconference room form

**Exceptions**

**exception** `indico.modules.vc.exceptions.VCRoomError` (*message, field=None*)

Bases: `Exception`

**exception** `indico.modules.vc.exceptions.VCRoomNotFoundError` (*message*)

Bases: `indico.modules.vc.exceptions.VCRoomError`

**7.1.29 Designer**

---

**Todo:** Docstrings (module, models, utilities)

---

**Models**

**class** `indico.modules.designer.models.images.DesignerImageFile` (*\*\*kwargs*)

Bases: `indico.core.storage.models.StoredFileMixin`, `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.



Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**content\_type**

The MIME type of the file.

**created\_dt**

The date/time when the file was uploaded.

**download\_url**

**extension**

The extension of the file.

**filename**

The name of the file.

**id**

The ID of the file

**locator**

**md5**

An MD5 hash of the file.

Automatically assigned when *save()* is called.

**size**

The size of the file (in bytes).

Automatically assigned when *save()* is called.

**storage\_backend**

**storage\_file\_id**

**template**

**template\_id**

The designer template the image belongs to

**version\_of = None**

```
class indico.modules.designer.models.templates.DesignerTemplate (**kwargs)
```

Bases: sqlalchemy.orm.decl\_api.Model

**background\_image**

**background\_image\_id**

**backside\_template**

**backside\_template\_id**

**category**

**category\_id**

**data**

**event**

**event\_id**

**id**

**is\_clonable**

**is\_system\_template**

**is\_ticket**

**locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

**owner**

**title**

**type**

## Utilities

`indico.modules.designer.util.get_all_templates(obj)`

Get all templates usable by an event/category.

`indico.modules.designer.util.get_badge_format(tpl)`

`indico.modules.designer.util.get_default_badge_on_category(category,`  
`only_inherited=False)`

`indico.modules.designer.util.get_default_ticket_on_category(category,`  
`only_inherited=False)`

`indico.modules.designer.util.get_image_placeholder_types()`

`indico.modules.designer.util.get_inherited_templates(obj)`

Get all templates inherited by a given event/category.

`indico.modules.designer.util.get_nested_placeholder_options()`

`indico.modules.designer.util.get_not_deletable_templates(obj)`

Get all non-deletable templates for an event/category.

`indico.modules.designer.util.get_placeholder_options()`

**class** `indico.modules.designer.pdf.DesignerPDFBase(template, config)`

Bases: `object`

`get_pdf()`

```
class indico.modules.designer.pdf.TplData(width, height, items, background_position,
                                         width_cm, height_cm)
    Bases: tuple
    Create new instance of TplData(width, height, items, background_position, width_cm, height_cm)

    background_position
        Alias for field number 3

    height
        Alias for field number 1

    height_cm
        Alias for field number 5

    items
        Alias for field number 2

    width
        Alias for field number 0

    width_cm
        Alias for field number 4
```

## Placeholders

```
class indico.modules.designer.placeholders.EventDatesPlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Event Dates'
    group = 'event'
    name = 'event_dates'

    classmethod render(event)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.

        Parameters kwargs – arguments specific to the placeholder’s context
```

```
class indico.modules.designer.placeholders.EventDescriptionPlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Event Description'
    group = 'event'
    name = 'event_description'

    classmethod render(event)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.

        Parameters kwargs – arguments specific to the placeholder’s context
```

```
class indico.modules.designer.placeholders.RegistrationFullNamePlaceholder
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name'

    name = 'full_name'

    name_options = {}

    with_title = True
```

```
class indico.modules.designer.placeholders.EventOrgTextPlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Event Organizers'

    group = 'event'

    name = 'event_organizers'

    classmethod render (event)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.
```

Parameters **kwargs** – arguments specific to the placeholder’s context

```
class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholder
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name (no title)'

    name = 'full_name_no_title'

    name_options = {}

    with_title = False
```

```
class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderB
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name B'

    name = 'full_name_b'

    name_options = {'last_name_first': False}

    with_title = True
```

```
class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderB
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name B (no title)'

    name = 'full_name_b_no_title'

    name_options = {'last_name_first': False}

    with_title = False
```

```
class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderC
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name C'

    name = 'full_name_c'
```

```

    name_options = {'last_name_first': False, 'last_name_upper': True}
    with_title = True

class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderC
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name C (no title)'
    name = 'full_name_no_title_c'
    name_options = {'last_name_upper': True}
    with_title = False

class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderD
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name D (abbrev.)'
    name = 'full_name_d'
    name_options = {'abbrev_first_name': True, 'last_name_first': False, 'last_name_upper': True}
    with_title = True

class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderD
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = 1'Full Name D (abbrev., no title)'
    name = 'full_name_no_title_d'
    name_options = {'abbrev_first_name': True, 'last_name_upper': True}
    with_title = False

class indico.modules.designer.placeholders.RegistrationTitlePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Title'
    field = 'title'
    name = 'title'

class indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = 1'First Name'
    field = 'first_name'
    name = 'first_name'

class indico.modules.designer.placeholders.RegistrationLastNamePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = 1'Last Name'
    field = 'last_name'
    name = 'last_name'

class indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Ticket QR Code'

```

```
group = 'registrant'
```

```
is_image = True
```

```
is_ticket = True
```

```
name = 'ticket_qr_code'
```

```
classmethod render(registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.designer.placeholders.RegistrationEmailPlaceholder
```

Bases: `indico.modules.designer.placeholders.RegistrationPlaceholder`

```
description = 1'E-mail'
```

```
field = 'email'
```

```
name = 'email'
```

```
class indico.modules.designer.placeholders.RegistrationAmountPlaceholder
```

Bases: `indico.modules.designer.placeholders.RegistrationPlaceholder`

```
description = 1'Price (no currency)'
```

```
name = 'amount'
```

```
classmethod render(registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.designer.placeholders.RegistrationPricePlaceholder
```

Bases: `indico.modules.designer.placeholders.RegistrationPlaceholder`

```
description = 1'Price (with currency)'
```

```
name = 'price'
```

```
classmethod render(registration)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.designer.placeholders.RegistrationFriendlyIDPlaceholder
```

Bases: `indico.modules.designer.placeholders.RegistrationPlaceholder`

```
description = 1'Registration ID'
```

```
field = 'friendly_id'
```

```

    name = 'registration_friendly_id'
class indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Institution'

    field = 'affiliation'

    name = 'affiliation'
class indico.modules.designer.placeholders.RegistrationPositionPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Position'

    field = 'position'

    name = 'position'
class indico.modules.designer.placeholders.RegistrationAddressPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Address'

    field = 'address'

    name = 'address'
class indico.modules.designer.placeholders.RegistrationCountryPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Country'

    field = 'country'

    name = 'country'
class indico.modules.designer.placeholders.RegistrationPhonePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = 1'Phone'

    field = 'phone'

    name = 'phone'
class indico.modules.designer.placeholders.EventTitlePlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Event Title'

    group = 'event'

    name = 'event_title'

    classmethod render(event)
        Convert the placeholder to a string.

        When a placeholder contains HTML that should not be escaped, the returned value should be returned as
        a markupsafe.Markup instance instead of a plain string.

        Subclasses are encouraged to explicitly specify the arguments they expect instead of using **kwargs.

        Parameters kwargs – arguments specific to the placeholder's context
class indico.modules.designer.placeholders.CategoryTitlePlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

```

```
description = l'Category Title'
```

```
group = 'event'
```

```
name = 'category_title'
```

```
classmethod render(event)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder's context

```
class indico.modules.designer.placeholders.EventRoomPlaceholder
```

Bases: `indico.modules.designer.placeholders.DesignerPlaceholder`

```
description = l'Event Room'
```

```
group = 'event'
```

```
name = 'event_room'
```

```
classmethod render(event)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder's context

```
class indico.modules.designer.placeholders.EventVenuePlaceholder
```

Bases: `indico.modules.designer.placeholders.DesignerPlaceholder`

```
description = l'Event Venue'
```

```
group = 'event'
```

```
name = 'event_venue'
```

```
classmethod render(event)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder's context

```
class indico.modules.designer.placeholders.EventSpeakersPlaceholder
```

Bases: `indico.modules.designer.placeholders.DesignerPlaceholder`

```
description = l'Event Speakers/Chairs'
```

```
group = 'event'
```

```
name = 'event_speakers'
```

```
classmethod render(event)
```

Convert the placeholder to a string.

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.



Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

```
class indico.modules.designer.placeholders.EventLogoPlaceholder
    Bases: indico.modules.designer.placeholders.DesignerPlaceholder

    description = 1'Event Logo'

    group = 'event'

    is_image = True

    name = 'event_logo'

    classmethod render (event)
        Convert the placeholder to a string.
```

When a placeholder contains HTML that should not be escaped, the returned value should be returned as a `markupsafe.Markup` instance instead of a plain string.

Subclasses are encouraged to explicitly specify the arguments they expect instead of using `**kwargs`.

**Parameters** `kwargs` – arguments specific to the placeholder’s context

### 7.1.30 Network

---

**Todo:** Docstrings (module, models)

---

#### Models

```
class indico.modules.networks.models.networks.IPNetwork (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

    group_id

    network
```

```
class indico.modules.networks.models.networks.IPNetworkGroup (**kwargs)
    Bases: sqlalchemy.orm.decl_api.Model

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

    attachment_access_override
        Grants all IPs in the network group read access to all attachments

    contains_ip (ip)

    description
```

**hidden**

Whether the network group is hidden in ACL forms

**id****locator****name****networks**

A descriptor that presents a read/write view of an object attribute.

**principal\_order** = 1

**principal\_type** = 5

## Utilities

`indico.modules.networks.util.serialize_ip_network_group(group)`

Serialize group to JSON-like object.

## 7.1.31 News

---

**Todo:** Docstrings (module, models)

---

## Models

**class** `indico.modules.news.models.news.NewsItem(**kwargs)`

Bases: `sqlalchemy.orm.decl_api.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

**anchor****content****created\_dt****id****locator**

Define a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```
slug
title
url
```

## Utilities

`indico.modules.news.util.get_recent_news()`  
Get a list of recent news for the home page.

## 7.1.32 Indico fields

---

**Todo:** Docstrings to all fields

---

Indico fields extend from WTForm fields and are used for the special cases where the simple form fields are not enough to cover all needs.

```
class indico.modules.events.fields.EventPersonLinkListField(*args, **kwargs)
    Bases: indico.modules.events.fields.PersonLinkListFieldBase

    A field to manage event's chairpersons.

    linked_object_attr = 'event'

    person_link_cls
        alias of indico.modules.events.models.persons.EventPersonLink

    pre_validate (form)
        Override if you need field-level validation. Runs before any other validators.

        Parameters form – The form the field belongs to.

    widget = <indico.web.forms.widgets.JinjaWidget object>

class indico.modules.events.fields.EventPersonListField(*args, **kwargs)
    Bases: indico.web.forms.fields.principals.PrincipalListField

    A field that lets you select a list Indico user and EventPersons.

    This requires its form to have an event set.

    create_untrusted_persons = False
        Whether new event persons created by the field should be marked as untrusted

    event
```

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**class** `indico.modules.events.fields.IndicoThemeSelectField(*args, **kwargs)`

Bases: `wtforms.fields.core.SelectField`

**class** `indico.modules.events.fields.PersonLinkListFieldBase(*args, **kwargs)`

Bases: `indico.modules.events.fields.EventPersonListField`

**default\_sort\_alpha** = `True`

If set to *True*, will be sorted alphabetically by default

**linked\_object\_attr** = `None`

name of the attribute on the form containing the linked object

**person\_link\_cls** = `None`

class that inherits from *PersonLinkBase*

**widget** = `None`

**class** `indico.modules.events.fields.RatingReviewField(*args, **kwargs)`

Bases: `wtforms.fields.core.RadioField`

**widget** = `<indico.web.forms.widgets.JinjaWidget object>`

**class** `indico.modules.events.fields.ReferencesField(*args, **kwargs)`

Bases: `indico.web.forms.fields.itemlists.MultipleItemsField`

A field to manage external references.

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**class** `indico.modules.events.abstracts.fields.AbstractField(*args, **kwargs)`

Bases: `wtforms.ext.sqlalchemy.fields.QuerySelectField`

A selectize-based field to select an abstract from an event.

**event**

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**search\_payload**

**search\_url**

**widget** = `<indico.web.forms.widgets.SelectizeWidget object>`

**class** `indico.modules.events.abstracts.fields.AbstractPersonLinkListField(*args,`

`**kwargs)`

Bases: `indico.modules.events.fields.PersonLinkListFieldBase`

A field to configure a list of abstract persons.

**create\_untrusted\_persons** = True

**default\_sort\_alpha** = False

**linked\_object\_attr** = 'abstract'

**person\_link\_cls**

alias of `indico.modules.events.abstracts.models.persons.AbstractPersonLink`

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.modules.events.abstracts.fields.EmailRuleListField(label=None,
                                                                validators=None,
                                                                filters=(),
                                                                description="",
                                                                id=None,
                                                                default=None,
                                                                widget=None,
                                                                render_kw=None,
                                                                _form=None,
                                                                _name=None,
                                                                _prefix="",
                                                                _translations=None,
                                                                _meta=None)
```

Bases: `indico.web.forms.fields.simple.JSONField`

A field that stores a list of e-mail template rules.

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.

- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**`CAN_POPULATE = True`**

**`accepted_condition_types = (<class 'indico.modules.events.abstracts.notifications.State`**

**`condition_choices`**

**`condition_class_map = {'contribution_type': <class 'indico.modules.events.abstracts.n`**

**`pre_validate(form)`**

Override if you need field-level validation. Runs before any other validators.

**Parameters** `form` – The form the field belongs to.

**`widget = <indico.web.forms.widgets.JinjaWidget object>`**

```
class indico.modules.events.abstracts.fields.TrackRoleField(label=None, val-
                                                                idators=None,
                                                                filters=(), descrip-
                                                                tion="", id=None,
                                                                default=None,
                                                                widget=None, ren-
                                                                der_kw=None,
                                                                _form=None,
                                                                _name=None,
                                                                _prefix="", _trans-
                                                                lations=None,
                                                                _meta=None)
```

Bases: `indico.web.forms.fields.simple.JSONField`

A field to assign track roles to principals.

Construct a new field.

#### Parameters

- **`label`** – The label of the field.
- **`validators`** – A sequence of validators to call when `validate` is called.
- **`filters`** – A sequence of filters which are run on input data by `process`.
- **`description`** – A description for the field, typically used for help text.
- **`id`** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **`default`** – The default value to assign to the field, if no form or object input is provided. May be a callable.

- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **\_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **\_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **\_meta** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If *\_form* and *\_name* isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**CAN\_POPULATE = True**

**category\_roles**

**event\_roles**

**permissions\_info**

**widget = <indico.web.forms.widgets.JinjaWidget object>**

```
class indico.modules.events.contributions.fields.ContributionPersonLinkListField(*args,
                                                                              **kwargs)
```

Bases: `indico.modules.events.fields.PersonLinkListFieldBase`

A field to configure a list of contribution persons.

**linked\_object\_attr = 'contrib'**

**person\_link\_cls**

alias of `indico.modules.events.contributions.models.persons.ContributionPersonLink`

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**widget = <indico.web.forms.widgets.JinjaWidget object>**

```
class indico.modules.events.contributions.fields.SubContributionPersonLinkListField(*args,
                                                                              **kwargs)
```

Bases: `indico.modules.events.contributions.fields.ContributionPersonLinkListField`

A field to configure a list of subcontribution persons.

**linked\_object\_attr = 'subcontrib'**

**person\_link\_cls**

alias of `indico.modules.events.contributions.models.persons.SubContributionPersonLink`

**widget = <indico.web.forms.widgets.JinjaWidget object>**

```
class indico.modules.events.papers.fields.PaperEmailSettingsField(label=None,
                                                                    validators=None,
                                                                    filters=(),
                                                                    description="",
                                                                    id=None,
                                                                    default=None,
                                                                    widget=None,
                                                                    render_kw=None,
                                                                    _form=None,
                                                                    _name=None,
                                                                    _prefix="",
                                                                    _translations=None,
                                                                    _meta=None)
```

Bases: `indico.web.forms.fields.simple.JSONField`

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **\_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **\_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **\_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *\_form* and *\_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.



**CAN\_POPULATE = True**

**event**

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.modules.events.sessions.fields.**SessionBlockPersonLinkListField** (\*args, \*\*kwargs)

Bases: *indico.modules.events.fields.PersonLinkListFieldBase*

**linked\_object\_attr** = 'session\_block'

**person\_link\_cls**

alias of *indico.modules.events.sessions.models.persons.SessionBlockPersonLink*

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.modules.categories.fields.**CategoryField** (\*args, \*\*kwargs)

Bases: *wtforms.fields.simple.HiddenField*

WTForms field that lets you select a category.

**Parameters** *require\_event\_creation\_rights* – Whether to allow selecting only categories where the user can create events.

**process\_data** (*value*)

Process the Python data applied to this field and store the result.

This will be called during form construction by the form's *kwargs* or *obj* argument.

**Parameters** *value* – The python object containing the value to process.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.modules.networks.fields.**MultiIPNetworkField** (\*args, \*\*kwargs)

Bases: *indico.web.forms.fields.itemlists.MultiStringField*

A field to enter multiple IPv4 or IPv6 networks.

The field data is a set of *IPNetwork*'s not bound to a DB session. The `unique` and `sortable` parameters of the parent class cannot be used with this class.

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**process\_data** (*value*)

Process the Python data applied to this field and store the result.

This will be called during form construction by the form's *kwargs* or *obj* argument.

**Parameters** *value* – The python object containing the value to process.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

```
class indico.web.forms.fields.IndicoSelectMultipleCheckboxField(label=None,  
                                                                valida-  
                                                                tors=None,  
                                                                coerce=<class  
                                                                'str'>,  
                                                                choices=None,  
                                                                vali-  
                                                                date_choice=True,  
                                                                **kwargs)
```

Bases: `wtforms.fields.core.SelectMultipleField`

**option\_widget** = `<wtforms.widgets.core.CheckboxInput object>`

**widget** = `<indico.web.forms.widgets.JinjaWidget object>`

```
class indico.web.forms.fields.IndicoRadioField(*args, **kwargs)
```

Bases: `wtforms.fields.core.RadioField`

**widget** = `<indico.web.forms.widgets.JinjaWidget object>`

```
class indico.web.forms.fields.JSONField(label=None, validators=None, filters=(), descrip-  
                                         tion="", id=None, default=None, widget=None,  
                                         render_kw=None, _form=None, _name=None,  
                                         _prefix="", _translations=None, _meta=None)
```

Bases: `wtforms.fields.simple.HiddenField`

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **\_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **\_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.

- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **\_meta** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**CAN\_POPULATE = False**

Whether an object may be populated with the data from this field

**populate\_obj** (*obj*, *name*)

Populates *obj*.<*name*> with the field’s data.

**Note** This is a destructive operation. If *obj*.<*name*> already exists, it will be overridden. Use with caution.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

```
class indico.web.forms.fields.HiddenFieldList (label=None, validators=None, filters=(),
                                              description="", id=None, default=None,
                                              widget=None, render_kw=None,
                                              _form=None, _name=None, _prefix="",
                                              _translations=None, _meta=None)
```

Bases: `wtforms.fields.simple.HiddenField`

A hidden field that handles lists of strings.

This is done *getlist*-style, i.e. by repeating the input element with the same name for each list item.

The only case where this field is useful is when you display a form via POST and provide a list of items (e.g. ids) related to the form which needs to be kept when the form is submitted and also need to access it via `request.form.getlist(...)` before submitting the form.

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.

- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**`process_formdata`** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**`widget`** = `<indico.web.forms.widgets.HiddenInputs object>`

```
class indico.web.forms.fields.TextListField(label=None, validators=None, filters=(),
                                             description="", id=None, default=None, widget=None, render_kw=None, _form=None,
                                             _name=None, _prefix="", _translations=None, _meta=None)
```

Bases: `wtforms.fields.simple.TextAreaField`

Construct a new field.

#### Parameters

- **`label`** – The label of the field.
- **`validators`** – A sequence of validators to call when *validate* is called.
- **`filters`** – A sequence of filters which are run on input data by *process*.
- **`description`** – A description for the field, typically used for help text.
- **`id`** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **`default`** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **`widget`** – If provided, overrides the widget used to render the field.
- **`render_kw`** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **`_form`** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.

- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **\_meta** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** **form** – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

```
class indico.web.forms.fields.EmailListField(label=None, validators=None, filters=(),
                                             description="", id=None, default=None,
                                             widget=None, render_kw=None,
                                             _form=None, _name=None, _prefix="",
                                             _translations=None, _meta=None)
```

Bases: `indico.web.forms.fields.simple.TextListField`

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **\_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **\_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.

- **\_meta** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If *\_form* and *\_name* isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

**class** `indico.web.forms.fields.IndicoPasswordField` (*\*args, \*\*kwargs*)

Bases: `wtforms.fields.simple.PasswordField`

Password field which can show or hide the password.

**widget** = `<indico.web.forms.widgets.PasswordWidget object>`

**class** `indico.web.forms.fields.IndicoStaticTextField` (*\*args, \*\*kwargs*)

Bases: `wtforms.fields.core.Field`

Return an html element with text taken from this field’s value.

**process\_data** (*data*)

Process the Python data applied to this field and store the result.

This will be called during form construction by the form’s *kwargs* or *obj* argument.

**Parameters value** – The python object containing the value to process.

**widget** = `<indico.web.forms.widgets.JinjaWidget object>`

**class** `indico.web.forms.fields.IndicoTagListField` (*label=None, validators=None, filters=(), description="", id=None, default=None, widget=None, render\_kw=None, \_form=None, \_name=None, \_prefix="", \_translations=None, \_meta=None*)

Bases: `indico.web.forms.fields.simple.HiddenFieldList`

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.

- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**`widget = <indico.web.forms.widgets.JinjaWidget object>`**

**`class indico.web.forms.fields.IndicoPalettePickerField(*args, **kwargs)`**

Bases: `indico.web.forms.fields.simple.JSONField`

Field allowing user to pick a color from a set of predefined values.

**`CAN_POPULATE = True`**

**`pre_validate(form)`**

Override if you need field-level validation. Runs before any other validators.

**Parameters** `form` – The form the field belongs to.

**`process_data(value)`**

Process the Python data applied to this field and store the result.

This will be called during form construction by the form’s `kwargs` or `obj` argument.

**Parameters** `value` – The python object containing the value to process.

**`process_formdata(valuelist)`**

Process data received over the wire from a form.

This will be called during form construction with data supplied through the `formdata` argument.

**Parameters** `valuelist` – A list of strings to process.

**`widget = <indico.web.forms.widgets.JinjaWidget object>`**

**`class indico.web.forms.fields.IndicoSinglePalettePickerField(*args, **kwargs)`**

Bases: `indico.web.forms.fields.colors.IndicoPalettePickerField`

Like `IndicoPalettePickerField` but for just a single color.

**`pre_validate(form)`**

Override if you need field-level validation. Runs before any other validators.

**Parameters** `form` – The form the field belongs to.

**`process_formdata(valuelist)`**

Process data received over the wire from a form.

This will be called during form construction with data supplied through the `formdata` argument.

**Parameters** `valuelist` – A list of strings to process.

**`class indico.web.forms.fields.TimeDeltaField(*args, **kwargs)`**

Bases: `wtforms.fields.core.Field`

A field that lets the user select a simple timedelta.

It does not support mixing multiple units, but it is smart enough to switch to a different unit to represent a timedelta that could not be represented otherwise.

**Parameters** **units** – The available units. Must be a tuple containing any any of ‘seconds’, ‘minutes’, ‘hours’ and ‘days’. If not specified, (‘hours’, ‘days’) is assumed.

**best\_unit**

Return the largest unit that covers the current timedelta.

**choices**

**magnitudes** = {‘days’: 86400, ‘hours’: 3600, ‘minutes’: 60, ‘seconds’: 1}

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** **form** – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

**unit\_names** = {‘days’: ‘Days’, ‘hours’: ‘Hours’, ‘minutes’: ‘Minutes’, ‘seconds’: ‘Seconds’}

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.web.forms.fields.IndicoDateTimeField(\*args, \*\*kwargs)

Bases: wtforms.ext.dateutil.fields.DateTimeField

Friendly datetime field that handles timezones and validations.

Important: When the form has a *timezone* field it must be declared before any *IndicoDateTimeField*. Otherwise its value is not available in this field resulting in an error during form submission.

**earliest\_dt**

**latest\_dt**

**linked\_datetime\_validator**

**linked\_field**

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** **form** – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

**timezone**

**timezone\_field**

**tzinfo**

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.web.forms.fields.OccurrencesField(\*args, \*\*kwargs)

Bases: indico.web.forms.fields.simple.JSONField



A field that lets you select multiple occurrences consisting of a start date/time and a duration.

**CAN\_POPULATE = True**

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

**timezone**

**timezone\_field**

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.web.forms.fields.IndicoTimezoneSelectField(\*args, \*\*kwargs)

Bases: wtforms.fields.core.SelectField

**process\_data** (*value*)

Process the Python data applied to this field and store the result.

This will be called during form construction by the form's *kwargs* or *obj* argument.

**Parameters value** – The python object containing the value to process.

**class** indico.web.forms.fields.IndicoEnumSelectField(*label=None, validators=None, enum=None, sorted=False, only=None, skip=None, none=None, titles=None, keep\_enum=True, \*\*kwargs*)

Bases: indico.web.forms.fields.enums.\_EnumFieldMixin, wtforms.fields.core.SelectFieldBase

Select field backed by a RichEnum.

**iter\_choices** ()

Provides data for choice widget rendering. Must return a sequence or iterable of (value, label, selected) tuples.

**widget** = <wtforms.widgets.core.Select object>

**class** indico.web.forms.fields.IndicoEnumRadioField(*label=None, validators=None, enum=None, sorted=False, only=None, skip=None, none=None, titles=None, keep\_enum=True, \*\*kwargs*)

Bases: indico.web.forms.fields.enums.IndicoEnumSelectField

**option\_widget** = <wtforms.widgets.core.RadioInput object>

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.web.forms.fields.HiddenEnumField(*label=None, validators=None, enum=None, only=None, skip=None, none=None, \*\*kwargs*)

Bases: indico.web.forms.fields.enums.\_EnumFieldMixin, wtforms.fields.simple.HiddenField

Hidden field that only accepts values from an Enum.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

```
class indico.web.forms.fields.FileField(*args, **kwargs)
```

Bases: wtforms.fields.core.Field

A dropzone field.

```
default_options = {'add_remove_links': True, 'handle_flashes': False, 'lightweight':
```

```
process_formdata(valuelist)
```

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.MultiStringField(*args, **kwargs)
```

Bases: wtforms.fields.simple.HiddenField

A field with multiple input text fields.

#### Parameters

- **field** – A tuple (fieldname, title) where the title is used in the placeholder.
- **uuid\_field** – If set, each item will have a UUID assigned and stored in the field specified here.
- **flat** – If True, the field returns a list of string values instead of dicts. Cannot be combined with *uuid\_field*.
- **unique** – Whether the values should be unique.
- **sortable** – Whether items should be sortable.

```
pre_validate(form)
```

Override if you need field-level validation. Runs before any other validators.

**Parameters** **form** – The form the field belongs to.

```
process_formdata(valuelist)
```

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** **valuelist** – A list of strings to process.

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.MultipleItemsField(*args, **kwargs)
```

Bases: wtforms.fields.simple.HiddenField

A field with multiple items consisting of multiple string values.

#### Parameters

- **fields** – A list of dicts with the following arguments: 'id': the unique ID of the field 'caption': the title of the column and the placeholder 'type': 'text|number|select', the type of the field 'coerce': callable to convert the value to a python type.

the type must be convertible back to a string, so usually you just want something like *int* or *float* here.

In case the type is ‘select’, the property ‘choices’ of the *MultipleItemsField* or the ‘choices’ kwarg needs to be a dict where the key is the ‘id’ of the select field and the value is another dict mapping the option’s id to its caption.

- **uuid\_field** – If set, each item will have a UUID assigned and stored in the field specified here. The name specified here may not be in *fields*.
- **uuid\_field\_opaque** – If set, the *uuid\_field* is considered opaque, i.e. it is never touched by this field. This is useful when you subclass the field and use e.g. actual database IDs instead of UUIDs.
- **unique\_field** – The name of a field in *fields* that needs to be unique.
- **sortable** – Whether items should be sortable.

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

**class** indico.web.forms.fields.OverrideMultipleItemsField(\*args, \*\*kwargs)

Bases: wtforms.fields.simple.HiddenField

A field similar to *MultipleItemsField* which allows the user to override some values.

**Parameters**

- **fields** – a list of (fieldname, title) tuples. Should match the fields of the corresponding *MultipleItemsField*.
- **field\_data** – the data from the corresponding *MultipleItemsField*.
- **unique\_field** – the name of the field which is unique among all rows
- **edit\_fields** – a set containing the field names which can be edited

If you decide to use this field, please consider adding support for *uuid\_field* here!

**get\_overridden\_value** (*row*, *name*)

Utility for the widget to get the entered value for an editable field.

**get\_row\_key** (*row*)

Utility for the widget to get the unique value for a row.

**pre\_validate** (*form*)

Override if you need field-level validation. Runs before any other validators.

**Parameters** *form* – The form the field belongs to.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.PrincipalListField(*args, **kwargs)
```

```
    Bases: wtforms.fields.simple.HiddenField
```

A field that lets you select a list of principals.

Principals are users or other objects representing users such as groups or roles that can be added to ACLs.

#### Parameters

- **allow\_external\_users** – If “search users with no indico account” should be available. Selecting such a user will automatically create a pending user once the form is submitted, even if other fields in the form fail to validate!
- **allow\_groups** – If groups should be selectable.
- **allow\_event\_roles** – If event roles should be selectable.
- **allow\_category\_roles** – If category roles should be selectable.
- **allow\_registration\_forms** – If registration form associated to an event should be selectable.
- **allow\_emails** – If the field should allow bare emails. Those are not selectable in the widget, but may be added to an ACL through other means.

```
process_formdata (valuelist)
```

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.PrincipalField(*args, **kwargs)
```

```
    Bases: wtforms.fields.simple.HiddenField
```

A field that lets you select a single Indico user.

**Parameters allow\_external\_users** – If “search users with no indico account” should be available. Selecting such a user will automatically create a pending user once the form is submitted, even if other fields in the form fail to validate!

```
process_formdata (valuelist)
```

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.AccessControlListField(*args, **kwargs)
```

```
    Bases: indico.web.forms.fields.principals.PrincipalListField
```

```
class indico.web.forms.fields.IndicoQuerySelectMultipleField(*args, **kwargs)
```

```
    Bases: wtforms.ext.sqlalchemy.fields.QuerySelectMultipleField
```

Like the parent, but with a callback that allows you to modify the list

The callback can return a new list or yield items, and you can use it e.g. to sort the list.

**data**

```
class indico.web.forms.fields.EditableFileField(*args, **kwargs)
```

```
    Bases: indico.web.forms.fields.files.FileField
```

A dropzone field that displays its current state and keeps track of deletes.

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoQuerySelectMultipleCheckboxField(*args,
                                                                    **kwargs)
    Bases: indico.web.forms.fields.sqlalchemy.IndicoQuerySelectMultipleField
```

**option\_widget** = <wtforms.widgets.core.CheckboxInput object>

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoLocationField(*args, **kwargs)
    Bases: indico.web.forms.fields.simple.JSONField
```

**CAN\_POPULATE** = True

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**widget** = <indico.web.forms.widgets.LocationWidget object>

```
class indico.web.forms.fields.IndicoMarkdownField(*args, **kwargs)
    Bases: wtforms.fields.simple.TextAreaField
```

A Markdown-enhanced textarea.

When using the editor you need to include the markdown JS/CSS bundles and also the MathJax JS bundle (even when using only the editor without MathJax).

**Parameters**

- **editor** – Whether to use the WMD widget with its live preview
- **mathjax** – Whether to use MathJax in the WMD live preview

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoDateField(*args, **kwargs)
    Bases: wtforms.ext.dateutil.fields.DateField
```

**earliest\_date**

**latest\_date**

**linked\_date\_validator**

**linked\_field**

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoProtectionField(*args, **kwargs)
    Bases: indico.web.forms.fields.enums.IndicoEnumRadioField
```

**radio\_widget** = <indico.web.forms.widgets.JinjaWidget object>

**render\_protection\_message** ()

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.IndicoSelectMultipleCheckboxBooleanField(label=None,
                                                                    val-
                                                                    ida-
                                                                    tors=None,
                                                                    co-
                                                                    erce=<class
                                                                    'str'>,
                                                                    choices=None,
                                                                    vali-
                                                                    date_choice=True,
                                                                    **kwargs)
```

Bases: `indico.web.forms.fields.simple.IndicoSelectMultipleCheckboxField`

**iter\_choices()**  
Provides data for choice widget rendering. Must return a sequence or iterable of (value, label, selected) tuples.

**process\_formdata(valuelist)**  
Process data received over the wire from a form.  
  
This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

```
class indico.web.forms.fields.RelativeDeltaField(*args, **kwargs)
```

Bases: `wtforms.fields.core.Field`

A field that lets the user select a simple timedelta.

It does not support mixing multiple units, but it is smart enough to switch to a different unit to represent a timedelta that could not be represented otherwise.

**Parameters units** – The available units. Must be a tuple containing any any of ‘seconds’, ‘minutes’, ‘hours’ and ‘days’. If not specified, (‘hours’, ‘days’) is assumed.

**choices**

```
magnitudes = {'days': relativedelta(days=+1), 'hours': relativedelta(hours=+1), 'min-
```

**pre\_validate(form)**  
Override if you need field-level validation. Runs before any other validators.

**Parameters form** – The form the field belongs to.

**process\_formdata(valuelist)**  
Process data received over the wire from a form.  
  
This will be called during form construction with data supplied through the *formdata* argument.

**Parameters valuelist** – A list of strings to process.

**split\_data**

```
unit_names = {'days': 'Days', 'hours': 'Hours', 'minutes': 'Minutes', 'months': 'M-
```

```
widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.IndicoWeekDayRepetitionField(*args, **kwargs)
```

Bases: `wtforms.fields.core.Field`

Field that lets you select an ordinal day of the week.

```
WEEK_DAY_NUMBER_CHOICES = ((1, 1'first'), (2, 1'second'), (3, 1'third'), (4, 1'fourth')
```

**day\_number\_data**

**process\_formdata** (*valuelist*)

Process data received over the wire from a form.

This will be called during form construction with data supplied through the *formdata* argument.

**Parameters** *valuelist* – A list of strings to process.

**week\_day\_data**

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoEmailRecipientsField(label=None,      val-
                                                         idators=None,      fil-
                                                         ters=(),  description="",
                                                         id=None,  default=None,
                                                         widget=None,      ren-
                                                         der_kw=None,
                                                         _form=None,
                                                         _name=None, _prefix="",
                                                         _translations=None,
                                                         _meta=None)
```

Bases: wtforms.fields.core.Field

Construct a new field.

#### Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render\_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **\_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **\_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **\_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **\_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **\_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If `_form` and `_name` isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

**process\_data** (*data*)

Process the Python data applied to this field and store the result.

This will be called during form construction by the form's *kwargs* or *obj* argument.

**Parameters value** – The python object containing the value to process.

**widget** = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoTimeField(label=None, validators=None, format='%H:%M', **kwargs)
```

Bases: `wtforms.fields.core.TimeField`

**widget** = <indico.web.forms.widgets.JinjaWidget object>



## 8.1 Changelog

### 8.1.1 Version 3.0.2

*Released on September 09, 2021*

#### Bugfixes

- Fix JavaScript errors on the login page which caused problems when using multiple form-based login methods (e.g. LDAP and local Indico accounts)

### 8.1.2 Version 3.0.1

*Released on September 08, 2021*

#### Improvements

- Allow filtering abstracts by custom fields having no value (#5033, #5034)
- Add support for syncing email addresses when logging in using external accounts (#5035)
- Use more space-efficient QR code version in registration tickets (#5052)
- Improve user experience when accessing an event restricted to registered participants while not logged in (#5053)
- When searching external users, prefer results with a name in case of multiple matches with the same email address (#5066)
- Show program codes in additional places (#5075)
- Display localized country names (#5070, #5076)

## Bugfixes

- Show correct placeholders in date picker fields (#5022)
- Correctly preselect the default currency when creating a registration form
- Do not notify registrants when a payment transaction is created in “pending” state
- Keep the order of multi-choice options in registration summary (#5020, #5032)
- Correctly handle relative URLs in PDF generation (#5042, #5044)
- Render markdown in track descriptions in PDF generation (#5043, #5044)
- Fix error when importing chairpersons from an existing event (#5047)
- Fix broken timetable entry permalinks when query string args are present (#5049)
- Do not show “Payments” event management menu entry for registration managers (#5072)
- Replace some hardcoded date formats with locale-aware ones (#5059, #5071)
- Clone the scientific program description together with tracks (#5077)
- Fix database error when importing registrations to an event that already contains a deleted registration form with registrations (#5078)

## Internal Changes

- Add `event.before_check_registration_email` signal (#5021, thanks @omegak)
- Do not strip image maps in places where HTML is allowed (#5026, thanks @bpedersen2)
- Add `event.registration.after_registration_form_clone` signal (#5037, thanks @vasantvohra)
- Add `registration-invite-options` template hook (#5045, thanks @vasantvohra)
- Fix Typeahead widget not working with extra validators (#5048, #5050, thanks @jbtwist)

## 8.1.3 Version 3.0

*Released on July 16, 2021*

## Major Features

- Add system notices which inform administrators about important things such as security problems or outdated Python/Postgres versions. These notices are retrieved once a day without sending any data related to the Indico instance, but if necessary, this feature can be disabled by setting `SYSTEM_NOTICES_URL` to `None` in `indico.conf` (#5004)
- It is now possible to use *SAML SSO* for authentication without the need for Shibboleth and Apache (#5014)

## Bugfixes

- Fix formatting and datetime localization in various PDF exports and timetable tab headers (#5009)
- Show lecture speakers as speakers instead of chairpersons on the participant roles page (#5008)

## Internal Changes

- Signals previously exposed directly via `signals.foo` now need to be accessed using their explicit name, i.e. `signals.core.foo` (#5007)
- Add `category.extra_events` signal (#5005, thanks @omegak)

## 8.1.4 Version 3.0rc2

*Released on July 09, 2021*

### Major Features

- Add support for personal tokens. These tokens act like OAuth tokens, but are associated with a specific user and generated manually without the need of doing the OAuth flow. They can be used like API keys but with better granularity using the same scopes OAuth applications have, and a single user can have multiple tokens using various scopes. By default any user can create such tokens, but admins can restrict their creation. (#1934, #4976)

### Improvements

- Add abstract content to the abstract list customization options (#4968)
- Add CLI option to create a series (#4969)
- Users cannot submit multiple anonymous surveys anymore by logging out and in again (#4693, #4970)
- Improve reviewing state display for paper reviewers (#4979, #4984)
- Make it clearer if the contributions/timetable of a conference are still in draft mode (#4977, #4986)
- Add “send to speakers” option in event reminders (#4958, #4966, thanks @Naveenaidu)
- Allow displaying all events descending from a category (#4982, #4983, thanks @omegak and @openprojects).
- Add an option to allow non-judge conveners to update an abstract track (#4989)

### Bugfixes

- Fix errors when importing events containing abstracts or event roles from a YAML dump (#4995)
- Fix sorting abstract notification rules (#4998)
- No longer silently fall back to the first event contact email address when sending registration emails where no explicit sender address has been configured (#4992, #4996, thanks @vasantvohra)
- Do not check for event access when using a registration link with a registration token (#4991, #4997, thanks @vasantvohra)

## 8.1.5 Version 3.0rc1

*Released on June 25, 2021*

## Major Features

- There is a new built-in search module which provides basic search functionality out of the box, and for more advanced needs (such as full text search in uploaded files) plugins can provide their own search functionality (e.g. using ElasticSearch). (#4841)
- Categories may now contain both events and subcategories at the same time. During the upgrade to 3.0 event creation is automatically set to restricted in all categories containing subcategories in order to avoid any negative surprises which would suddenly allow random Indico users to create events in places where they couldn't do so previously. (#4679, #4725, #4757)
- The OAuth provider module has been re-implemented based on a more modern library (authlib). Support for the somewhat insecure *implicit flow* has been removed in favor of the code-with-PKCE flow. Tokens are now stored more securely as a hash instead of plaintext. For a given user/app/scope combination, only a certain amount of tokens are stored; once the limit has been reached older tokens will be discarded. The OAuth provider now exposes its metadata via a well-known URI (RFC 8414) and also has endpoints to introspect or revoke a token. (#4685, #4798)
- User profile pictures (avatars) are now shown in many more places throughout Indico, such as user search results, meeting participant lists and reviewing timelines. (#4625, #4747, #4939)

## Internationalization

- New locale: English (United States)
- New translation: Turkish

## Improvements

- Use a more modern search dialog when searching for users (#4674, #4743)
- Add an option to refresh event person data from the underlying user when cloning an event (#4750, #4760)
- Add options for attaching iCal files to complete registration and event reminder emails (#1158, #4780)
- Use the new token-based URLs instead of API keys for persistent ical links and replace the calendar link widgets in category, event, session and contribution views with the more modern ones used in dashboard (#4776, #4801)
- Add an option to export editables to JSON (#4767, #4810)
- Add an option to export paper peer reviewing data to JSON (#4767, #4818)
- Passwords are now checked against a list of breached passwords (“Have I Been Pwned”) in a secure and anonymous way that does not disclose any data. If a user logs in with an insecure password, they are forced to change it before they can continue using Indico (#4817)
- Failed login attempts now trigger rate limiting to prevent brute-force attacks (#1550, #4817)
- Allow filtering the “Participant Roles” page by users who have not registered for the event (#4763, #4822)
- iCalendar exports now include contact data, event logo URL and, when exporting sessions/contributions, the UID of the related event. Also, only non-empty fields are exported. (#4785, #4586, #4587, #4791, #4820)
- Allow adding groups/roles as “authorized abstract submitters” (#4834)
- Direct links to (sub-)contributions in meetings using the URLs usually meant for conferences now redirect to the meeting view page (#4847)
- Use a more compact setup QR code for the mobile *Indico check-in* app; the latest version of the app is now required. (#4844)

- Contribution duration fields now use a widget similar to the time picker that makes selecting durations easier. (#2462, #4873)
- Add new meeting themes that show sequential numbers instead of start times for contributions (#4899)
- Remove the very outdated “Compact style” theme (it’s still available via the `themes_legacy` plugin) (#4900, #4899)
- Support cloning surveys when cloning events (#2045, #4910)
- Show external contribution references in conferences (#4928, #4933)
- Allow changing the rating scale in abstract/paper reviewing even after reviewing started (#4942)
- Allow blacklisting email addresses for user registrations (#4644, #4946)

## Bugfixes

- Take registrations of users who are only members of a custom event role into account on the “Participant Roles” page (#4822)
- Fail gracefully during registration import when two rows have different emails that belong to the same user (#4823)
- Restore the ability to see who’s inheriting access from a parent object (#4833)
- Fix misleading message when cancelling a booking that already started and has past occurrences that won’t be cancelled (#4719, #4861)
- Correctly count line breaks in length-limited abstracts (#4918)
- Fix error when trying to access subcontributions while event is in draft mode
- Update the user link in registrations when merging two users (#4936)
- Fix error when exporting a conference timetable PDF with the option “Print abstract content of all contributions” and one of the abstracts is too big to fit in a page (#4881, #4955)
- Emails sent via the Editing module are now logged to the event log (#4960)
- Fix error when importing event notes from another event while the target event already has a deleted note (#4959)

## Internal Changes

- Require Python 3.9 - older Python versions (especially Python 2.7) are **no longer supported**
- `confId` has been changed to `event_id` and the corresponding URL path segments now enforce numeric data (and thus pass the id as a number instead of string)
- `CACHE_BACKEND` has been removed; Indico now always uses Redis for caching
- The integration with flower (celery monitoring tool) has been removed as it was not widely used, did not provide much benefit, and it is no longer compatible with the latest Celery version
- `session.user` now returns the user related to the current request, regardless of whether it’s coming from OAuth, a signed url or the actual session (#4803)
- Add a new `check_password_secure` signal that can be used to implement additional password security checks (#4817)
- Add an endpoint to let external applications stage the creation of an event with some data to be pre-filled when the user then opens the link returned by that endpoint (#4628, thanks @adl1995)

## 8.1.6 Version 2.3.6

*Unreleased*

### Bugfixes

- None so far :)

## 8.1.7 Version 2.3.5

*Released on May 11, 2021*

### Security fixes

- Fix XSS vulnerabilities in the category picker (via category titles), location widget (via room and venue names defined by an Indico administrator) and the “Indico Weeks View” timetable theme (via contribution/break titles defined by an event organizer). As neither of these objects can be created by untrusted users (on a properly configured instance) we consider the severity of this vulnerability “minor” (#4897)

### Internationalization

- New translation: Polish
- New translation: Mongolian

### Improvements

- Add an option to not disclose the names of editors and commenters to submitters in the Paper Editing module (#4829, #4865)

### Bugfixes

- Do not show soft-deleted long-lasting events in category calendar (#4824)
- Do not show management-related links in editing hybrid view unless the user has access to them (#4830)
- Fix error when assigning paper reviewer roles with notifications enabled and one of the reviewing types disabled (#4838)
- Fix viewing timetable entries if you cannot access the event but a specific session inside it (#4857)
- Fix viewing contributions if you cannot access the event but have explicit access to the contribution (#4860)
- Hide registration menu item if you cannot access the event and registrations are not exempt from event access checks (#4860)
- Fix inadvertently deleting a file uploaded during the “make changes” Editing action, resulting in the revision sometimes still referencing the file even though it has been deleted from storage (#4866)
- Fix sorting abstracts by date (#4877)

## Internal Changes

- Add `before_notification_send` signal (#4874, thanks @omegak)

## 8.1.8 Version 2.3.4

*Released on March 11, 2021*

### Security fixes

- Fix some open redirects which could help making harmful URLs look more trustworthy by linking to Indico and having it redirect the user to a malicious site (#4814, #4815)
- The `BASE_URL` is now always enforced and requests whose Host header does not match are rejected. This prevents malicious actors from tricking Indico into sending e.g. a password reset link to a user that points to a host controlled by the attacker instead of the actual Indico host (#4815)

---

**Note:** If the webserver is already configured to enforce a canonical host name and redirects or rejects such requests, this cannot be exploited. Additionally, exploiting this problem requires user interaction: they would need to click on a password reset link which they never requested, and which points to a domain that does not match the one where Indico is running.

---

### Improvements

- Fail more gracefully if a user has an invalid locale set and fall back to the default locale or English in case the default locale is invalid as well
- Log an error if the configured default locale does not exist
- Add ID-1 page size for badge printing (#4774, thanks @omegak)
- Allow managers to specify a reason when rejecting registrants and add a new placeholder for the rejection reason when emailing registrants (#4769, thanks @vasantvohra)

### Bugfixes

- Fix the “Videoconference Rooms” page in conference events when there are any VC rooms attached but the corresponding plugin is no longer installed
- Fix deleting events which have a videoconference room attached which has its VC plugin no longer installed
- Do not auto-redirect to SSO when an MS office user agent is detected (#4720, #4731)
- Allow Editing team to view editables of unpublished contributions (#4811, #4812)

## Internal Changes

- Also trigger the `ical-export` metadata signal when exporting events for a whole category
- Add `primary_email_changed` signal (#4802, thanks @openprojects)

### 8.1.9 Version 2.3.3

*Released on January 25, 2021*

#### Security fixes

- JSON locale data for invalid locales is no longer cached on disk; instead a 404 error is triggered. This avoids creating small files in the cache folder for each invalid locale that is requested. (#4766)

#### Internationalization

- New translation: Ukrainian

#### Improvements

- Add a new “Until approved” option for a registration form’s “Modification allowed” setting (#4740, thanks @vasantvohra)
- Show last login time in dashboard (#4735, thanks @vasantvohra)
- Allow Markdown in the “Message for complete registrations” option of a registration form (#4741)
- Improve video conference linking dropdown for contributions/sessions (hide unscheduled, show start time) (#4753)
- Show timetable filter button in conferences with a meeting-like timetable

#### Bugfixes

- Fix error when converting malformed HTML links to LaTeX
- Hide inactive contribution/abstract fields in submit/edit forms (#4755)
- Fix adding registrants to a session ACL

#### Internal Changes

- Videoconference plugins may now display a custom message for the prompt when deleting a videoconference room (#4733)
- Videoconference plugins may now override the behavior when cloning an event with attached videoconference rooms (#4732)

### 8.1.10 Version 2.3.2

*Released on November 30, 2020*



## Improvements

- Disable title field by default in new registration forms (#4688, #4692)
- Add gender-neutral “Mx” title (#4688, #4692)
- Add contributions placeholder for emails (#4716, thanks @bpedersen2)
- Show program codes in contribution list (#4713)
- Display the target URL of link materials if the user can access them (#2599, #4718)
- Show the revision number for all revisions in the Editing timeline (#4708)

## Bugfixes

- Only consider actual speakers in the “has registered speakers” contribution list filter (#4712, thanks @bpedersen2)
- Correctly filter events in “Sync with your calendar” links (this fix only applies to newly generated links) (#4717)
- Correctly grant access to attachments inside public sessions/contribs even if the event is more restricted (#4721)
- Fix missing filename pattern check when suggesting files from Paper Peer Reviewing to submit for Editing (#4715)
- Fix filename pattern check in Editing when a filename contains dots (#4715)
- Require explicit admin override (or being whitelisted) to override blockings (#4706)
- Clone custom abstract/contribution fields when cloning abstract settings (#4724, thanks @bpedersen2)
- Fix error when rescheduling a survey that already has submissions (#4730)

### 8.1.11 Version 2.3.1

*Released on October 27, 2020*

## Security fixes

- Fix potential data leakage between OAuth-authenticated and unauthenticated HTTP API requests for the same resource (#4663)

---

**Note:** Due to OAuth access to the HTTP API having been broken until this version, we do not believe this was actually exploitable on any Indico instance. In addition, only Indico administrators can create OAuth applications, so regardless of the bug there is no risk for any instance which does not have OAuth applications with the `read:legacy_api` scope.

---

## Improvements

- Generate material packages in a background task to avoid timeouts or using excessive amounts of disk space in case of people submitting several times (#4630)
- Add new `EXPERIMENTAL_EDITING_SERVICE` setting to enable extending an event’s Editing workflow through an OpenReferee server (#4659)

## Bugfixes

- Only show the warning about draft mode in a conference if it actually has any contributions or timetable entries
- Do not show incorrect modification deadline in abstract management area if no such deadline has been set (#4650)
- Fix layout problem when minutes contain overly large embedded images (#4653, #4654)
- Prevent pending registrations from being marked as checked-in (#4646, thanks @omegak)
- Fix OAuth access to HTTP API (#4663)
- Fix ICS export of events with draft timetable and contribution detail level (#4666)
- Fix paper revision submission field being displayed for judges/reviewers (#4667)
- Fix managers not being able to submit paper revisions on behalf of the user (#4667)

## Internal Changes

- Add `registration_form_wtform_created` signal and send form data in `registration_created` and `registration_updated` signals (#4642, thanks @omegak)
- Add `logged_in` signal

## 8.1.12 Version 2.3

*Released on September 14, 2020*

---

**Note:** We also published a [blog post](#) summarizing the most relevant changes for end users.

---

## Major Features

- Add category roles, which are similar to local groups but within the scope of a category and its subcategories. They can be used for assigning permissions in any of these categories and events within such categories.
- Events marked as “Invisible” are now hidden from the category’s event list for everyone except managers (#4419, thanks @openprojects)
- Introduce profile picture, which is for now only visible on the user dashboard (#4431, thanks @omegak)
- Registrants can now be added to event ACLs. This can be used to easily restrict parts of an event to registered participants. If registration is open and a registration form is in the ACL, people will be able to access the registration form even if they would otherwise not have access to the event itself. It is also possible to restrict individual event materials and custom page/link menu items to registered participants. (#4477, #4528, #4505, #4507)
- Add a new Editing module for papers, slides and posters which provides a workflow for having a team review the layout/formatting of such proceedings and then publish the final version on the page of the corresponding contribution. The Editing module can also be connected to an external microservice to handle more advanced workflows beyond what is supported natively by Indico.

## Internationalization

- New translation: Chinese (Simplified)

## Improvements

- Sort survey list by title (#3802)
- Hide “External IDs” field if none are defined (#3857)
- Add LaTeX source export for book of abstracts (#4035, thanks @bpedersen2)
- Tracks can now be categorized in track groups (#4052)
- Program codes for sessions, session blocks, contributions and subcontributions can now be auto-generated (#4026)
- Add draft mode for the contribution list of conference events which hides pages like the contribution list and timetable until the event organizers publish the contribution list. (#4095)
- Add ICS export for information in the user dashboard (#4057)
- Allow data syncing with multipass providers which do not support refreshing identity information
- Show more verbose error when email validation fails during event registration (#4177)
- Add link to external map in room details view (#4146)
- Allow up to 9 digits (instead of 6) before the decimal point in registration fees
- Add button to booking details modal to copy direct link (#4230)
- Do not require new room manager approval when simply shortening a booking (#4214)
- Make root category description/title customizable using the normal category settings form (#4231)
- Added new `LOCAL_GROUPS` setting that can be used to fully disable local groups (#4260)
- Log bulk event category changes in the event log (#4241)
- Add CLI commands to block and unblock users (#3845)
- Show warning when trying to merge a blocked user (#3845)
- Allow importing event role members from a CSV file (#4301)
- Allow optional comment when accepting a pre-booking (#4086)
- Log event restores in event log (#4309)
- Warn about cancelling/rejecting whole recurring bookings instead of just specific occurrences (#4092)
- Add “quick cancel” link to room booking reminder emails (#4324)
- Add visual information and filtering options for participants’ registration status to the contribution list (#4318)
- Add warning when accepting a pre-booking in case there are concurrent bookings (#4129)
- Add event logging to opening/closing registration forms, approval/rejection of registrations, and updates to event layout (#4360, thanks @giusedb & @omegak)
- Add category navigation dialog on category display page (#4282, thanks @omegak)
- Add UI for admins to block/unblock users (#3243)
- Show labels indicating whether a user is an admin, blocked or soft-deleted (#4363)
- Add map URL to events, allowing also to override room map URL (#4402, thanks @omegak)

- Use custom time picker for time input fields taking into account the 12h/24h format of the user's locale (#4399)
- Refactor the room edit modal to a tabbed layout and improve error handling (#4408)
- Preserve non-ascii characters in file names (#4465)
- Allow resetting moderation state from registration management view (#4498, thanks @omegak)
- Allow filtering event log by related entries (#4503, thanks @omegak)
- Do not automatically show the browser's print dialog in a meeting's print view (#4513)
- Add "Add myself" button to person list fields (e.g. for abstract authors) (#4411, thanks @jgrigera)
- Subcontributions can now be managed from the meeting display view (#2679, #4520)
- Add CfA setting to control whether authors can edit abstracts (#3431)
- Add CfA setting to control whether only speakers or also authors should get submission rights once the abstract gets accepted (#3431)
- Show the Indico version in the footer again (#4558)
- Event managers can upload a custom Book of Abstract PDF (#3039, #4577)
- Display each news item on a separate page instead of together with all the other news items (#4587)
- Allow registrants to withdraw their application (#2715, #4585, thanks @brabemi & @omegak)
- Allow choosing a default badge in categories (#4574, thanks @omegak)
- Display event labels on the user's dashboard as well (#4592)
- Event modules can now be imported from another event (#4518, thanks @meluru)
- Event modules can now be imported from another event (#4518, #4533, thanks @meluru)
- Include the event keywords in the event API data (#4598, #4599, thanks @chernals)
- Allow registrants to check details for non-active registrations and prevent them from registering twice with the same registration form (#4594, #4595, thanks @omegak)
- Add a new `CUSTOM_LANGUAGES` setting to `indico.conf` to override the name/territory of a language or disable it altogether (#4620)

## Bugfixes

- Hide Book of Abstracts menu item if LaTeX is disabled and no custom Book of Abstracts has been uploaded
- Use a more consistent order when cloning the timetable (#4227)
- Do not show unrelated rooms with similar names when booking room from an event (#4089)
- Stop icons from overlapping in the datetime widget (#4342)
- Fix alignment of materials in events (#4344)
- Fix misleading wording in protection info message (#4410)
- Allow guests to access public notes (#4436)
- Allow width of weekly event overview table to adjust to window size (#4429)
- Fix whitespace before punctuation in Book of Abstracts (#4604)
- Fix empty entries in corresponding authors (#4604)
- Actually prevent users from editing registrations if modification is disabled

- Handle LaTeX images with broken redirects (#4623, thanks @bcc)

### Internal Changes

- Make React and SemanticUI usable everywhere (#3955)
- Add before-regform template hook (#4171, thanks @giusedb)
- Add registrations kwarg to the event.designer.print\_badge\_template signal (#4297, thanks @giusedb)
- Add registration\_form\_edited signal (#4421, thanks @omegak)
- Make PyIntEnum freeze enums in Alembic revisions (#4425, thanks @omegak)
- Add before-registration-summary template hook (#4495, thanks @omegak)
- Add extra-registration-actions template hook (#4500, thanks @omegak)
- Add event-management-after-title template hook (#4504, thanks @meluru)
- Save registration id in related event log entries (#4503, thanks @omegak)
- Add before-registration-actions template hook (#4524, thanks @omegak)
- Add LinkedDate and DateRange form field validators (#4535, thanks @omegak)
- Add extra-regform-settings template hook (#4553, thanks @meluru)
- Add filter\_selectable\_badges signal (#4557, thanks @omegak)
- Add user ID in every log record logged in a request context (#4570, thanks @omegak)
- Add extra-registration-settings template hook (#4596, thanks @meluru)
- Allow extending polymorphic models in plugins (#4608, thanks @omegak)
- Wrap registration form AngularJS directive in jinja block for more easily overriding arguments passed to the app in plugins (#4624, thanks @omegak)

---

### 8.1.13 Version 2.2.9

*Unreleased*

#### Bugfixes

- Fix error when building LaTeX PDFs if the temporary event logo path contained an underscore (#4521)
- Disallow storing invalid timezones in user settings and reduce risk of sending wrong timezone names when people automatically translate their UI (#4529)

### 8.1.14 Version 2.2.8

*Released on April 08, 2020*

## Security fixes

- Update [bleach](#) to fix a regular expression denial of service vulnerability
- Update [Pillow](#) to fix a buffer overflow vulnerability

### 8.1.15 Version 2.2.7

*Released on March 23, 2020*

## Improvements

- Add support for event labels to indicate e.g. postponed or cancelled events ([#3199](#))

## Bugfixes

- Allow slashes in roomName export API
- Show names instead of IDs of local groups in ACLs ([#3700](#))

### 8.1.16 Version 2.2.6

*Released on February 27, 2020*

## Bugfixes

- Fix some email fields (error report contact, agreement cc address) being required even though they should be optional
- Avoid browsers prefilling stored passwords in togglable password fields such as the event access key
- Make sure that tickets are not attached to emails sent to registrants for whom tickets are blocked ([#4242](#))
- Fix event access key prompt not showing when accessing an attachment link ([#4255](#))
- Include event title in OpenGraph metadata ([#4288](#))
- Fix error when viewing abstract with reviews that have no scores
- Update requests and pin idna to avoid installing incompatible dependency versions ([#4327](#))

### 8.1.17 Version 2.2.5

*Released on December 06, 2019*

## Improvements

- Sort posters in timetable PDF export by board number ([#4147](#), thanks [@bpedersen2](#))
- Use lat/lng field order instead of lng/lat when editing rooms ([#4150](#), thanks [@bpedersen2](#))
- Add additional fields to the contribution csv/xlsx export (authors and board number) ([#4148](#), thanks [@bpedersen2](#))

## Bugfixes

- Update the Pillow library to 6.2.1. This fixes an issue where some malformed images could result in high memory usage or slow processing.
- Truncate long speaker names in the timetable instead of hiding them (#4110)
- Fix an issue causing errors when using translations for languages with no plural forms (like Chinese).
- Fix creating rooms without touching the longitude/latitude fields (#4115)
- Fix error in HTTP API when Basic auth headers are present (#4123, thanks @uxmaster)
- Fix incorrect font size in some room booking dropdowns (#4156)
- Add missing email validation in some places (#4158)
- Reject requests containing NUL bytes in the POST data (#4159)
- Fix truncated timetable PDF when using “Print each session on a separate page” in an event where the last timetable entry of the day is a top-level contribution or break (#4134, thanks @bpedersen2)
- Only show public contribution fields in PDF exports (#4165)
- Allow single arrival/departure date in accommodation field (#4164, thanks @bpedersen2)

### 8.1.18 Version 2.2.4

*Released on October 16, 2019*

#### Security fixes

- Fix more places where LaTeX input was not correctly sanitized. While the biggest security impact (reading local files) has already been mitigated when fixing the initial vulnerability in the previous release, it is still strongly recommended to update.

### 8.1.19 Version 2.2.3

*Released on October 08, 2019*

#### Security fixes

- Strip @, +, - and = from the beginning of strings when exporting CSV files to avoid security issues when opening the CSV file in Excel
- Use 027 instead of 000 umask when temporarily changing it to get the current umask
- Fix LaTeX sanitization to prevent malicious users from running unsafe LaTeX commands through specially crafted abstracts or contribution descriptions, which could lead to the disclosure of local file contents

## Improvements

- Improve room booking interface on small-screen devices (#4013)
- Add user preference for room owners/manager to select if they want to receive notification emails for their rooms (#4096, #4098)

- Show family name field first in user search dialog (#4099)
- Make date headers clickable in room booking calendar (#4099)
- Show times in room booking log entries (#4099)
- Support disabling server-side LaTeX altogether and hide anything that requires it (such as contribution PDF export or the Book of Abstracts). **LaTeX is now disabled by default, unless the `XELATEX_PATH` is explicitly set in `indico.conf`.**

### Bugfixes

- Remove 30s timeout from dropzone file uploads
- Fix bug affecting room booking from an event in another timezone (#4072)
- Fix error when commenting on papers (#4081)
- Fix performance issue in conferences with public registration count and a high amount of registrations
- Fix confirmation prompt when disabling conference menu customizations (#4085)
- Fix incorrect days shown as weekend in room booking for some locales
- Fix ACL entries referencing event roles from the old event when cloning an event with event roles in the ACL. Run `indico maint fix-event-role-acls` after updating to fix any affected ACLs (#4090)
- Fix validation issues in coordinates fields when editing rooms (#4103)

## 8.1.20 Version 2.2.2

*Released on August 23, 2019*

### Bugfixes

- Remove dependency on `pyatom`, which has vanished from PyPI

## 8.1.21 Version 2.2.1

*Released on August 16, 2019*

### Improvements

- Make list of event room bookings sortable (#4022)
- Log when a booking is split during editing (#4031)
- Improve “Book” button in multi-day events (#4021)

### Bugfixes

- Add missing slash to the `template_prefix` of the `designer` module
- Always use HH:MM time format in book-from-event link
- Fix timetable theme when set to “indico weeks view” before 2.2 (#4027)



- Avoid flickering of booking edit details tooltip
- Fix outdated browser check on iOS (#4033)

## 8.1.22 Version 2.2

*Released on August 06, 2019*

### Major Changes

- **Drop support for Internet Explorer 11 and other outdated or discontinued browser versions.** Indico shows a warning message when accessed using such a browser. The latest list of supported browsers can be found [in the README on GitHub](#), but generally Indico now supports the last two versions of each major browser (determined at release time), plus the current Firefox ESR.
- Rewrite the room booking frontend to be more straightforward and user-friendly. Check [our blog](#) for details.

### Improvements

- Rework the event log viewer to be more responsive and not freeze the whole browser when there are thousands of log entries
- Add shortcut to next upcoming event in a category (#3388)
- Make registration period display less confusing (#3359)
- Add edit button to custom conference pages (#3284)
- Support markdown in survey questions (#3366)
- Improve event list in case of long event titles (#3607, thanks @nop33)
- Include event page title in the page's <title> (#3285, thanks @bpedersen2)
- Add option to include subcategories in upcoming events (#3449)
- Allow event managers to override the name format used in the event (#2455)
- Add option to not clone venue/room of an event
- Show territory/country next to the language name (#3968)
- Add more sorting options to book of abstracts (#3429, thanks @bpedersen2)
- Add more formatting options to book of abstracts (#3335, thanks @bpedersen2)
- Improve message when the call for abstracts is scheduled to open but hasn't started yet
- Make link color handling for LaTeX pdfs configurable (#3283, thanks @bpedersen2)
- Preserve displayed order in contribution exports that do not apply any specific sorting (#4005)
- Add author list button to list of papers (#3978)

### Bugfixes

- Fix incorrect order of session blocks inside timetable (#2999)
- Add missing email validation to contribution CSV import (#3568, thanks @Kush22)
- Do not show border after last item in badge designer toolbar (#3607, thanks @nop33)

- Correctly align centered footer links (#3599, thanks @nop33)
- Fix top/right alignment of session bar in event display view (#3599, thanks @nop33)
- Fix error when trying to create a user with a mixed-case email address in the admin area
- Fix event import if a user in the exported data has multiple email addresses and they match different users
- Fix paper reviewers getting notifications even if their type of reviewing has been disabled (#3852)
- Correctly handle merging users in the paper reviewing module (#3895)
- Show correct number of registrations in management area (#3935)
- Fix sorting book of abstracts by board number (#3429, thanks @bpedersen2)
- Enforce survey submission limit (#3256)
- Do not show “Mark as paid” button and checkout link while a transaction is pending (#3361, thanks @driehle)
- Fix 404 error on custom conference pages that do not have any ascii chars in the title (#3998)
- Do not show pending registrants in public participant lists (#4017)

### Internal Changes

- Use webpack to build static assets
  - Add React+Redux for new frontend modules
  - Enable modern ES201x features
- 

## 8.1.23 Version 2.1.11

*Released on October 16, 2019*

### Security fixes

- Fix more places where LaTeX input was not correctly sanitized. While the biggest security impact (reading local files) has already been mitigated when fixing the initial vulnerability in the previous release, it is still strongly recommended to update.

## 8.1.24 Version 2.1.10

*Released on October 08, 2019*

### Security fixes

- Strip @, +, - and = from the beginning of strings when exporting CSV files to avoid security issues when opening the CSV file in Excel
- Use 027 instead of 000 umask when temporarily changing it to get the current umask
- Fix LaTeX sanitization to prevent malicious users from running unsafe LaTeX commands through specially crafted abstracts or contribution descriptions, which could lead to the disclosure of local file contents

### 8.1.25 Version 2.1.9

*Released on August 26, 2019*

#### Bugfixes

- Fix bug in calendar view, due to timezones (#3903)
- Remove dependency on `pyatom`, which has vanished from PyPI (#4045)

### 8.1.26 Version 2.1.8

*Released on March 12, 2019*

#### Improvements

- Add A6 to page size options (#3793)

#### Bugfixes

- Fix celery/redis dependency issue (#3809)

### 8.1.27 Version 2.1.7

*Released on January 24, 2019*

#### Improvements

- Add setting for the default contribution duration of an event (#3446)
- Add option to copy abstract attachments to contributions when accepting them (#3732)

#### Bugfixes

- Really fix the oauthlib conflict (was still breaking in some cases)

### 8.1.28 Version 2.1.6

*Released on January 15, 2019*

#### Bugfixes

- Allow adding external users as speakers/chairpersons (#3562)
- Allow adding external users to event ACLs (#3562)
- Pin requests-oauthlib version to avoid dependency conflict

### 8.1.29 Version 2.1.5

*Released on December 06, 2018*

#### Improvements

- Render the reviewing state of papers in the same way as abstracts (#3665)

#### Bugfixes

- Use correct speaker name when exporting contributions to spreadsheets
- Use friendly IDs in abstract attachment package folder names
- Fix typo in material package subcontribution folder names
- Fix check on whether registering for an event is possible
- Show static text while editing registrations (#3682)

### 8.1.30 Version 2.1.4

*Released on September 25, 2018*

#### Bugfixes

- Let managers download tickets for registrants even if all public ticket downloads are disabled (#3493)
- Do not count deleted registrations when printing tickets from the badge designer page
- Hide “Save answers” in surveys while not logged in
- Fix importing event archives containing registrations with attachments
- Fix display issue in participants table after editing data (#3511)
- Fix errors when booking rooms via API

### 8.1.31 Version 2.1.3

*Released on August 09, 2018*

#### Security fixes

- Only return timetable entries for the current session when updating a session through the timetable (#3474, thanks @glunardi for reporting)
- Prevent session managers/coordinators from modifying certain timetable entries or scheduling contributions not assigned to their session
- Restrict access to timetable entry details to users who are authorized to see them

## Improvements

- Improve survey result display (#3486)
- Improve email validation for registrations (#3471)

## Bugfixes

- Point to correct day in “edit session timetable” link (#3419)
- Fix error when exporting abstracts with review questions to JSON
- Point the timetable to correct day in the session details
- Fix massive performance issue on the material package page in big events
- Fix error when using the checkin app to mark someone as checked in (#3473, thanks @femtobit)
- Fix error when a session coordinator tries changing the color of a break using the color picker in the balloon’s tooltip

## Internal Changes

- Add some new signals and template hooks to the registration module

## 8.1.32 Version 2.1.2

*Released on June 11, 2018*

## Improvements

- Show email address for non-anonymous survey submissions (#3258)

## Bugfixes

- Show question description in survey results (#3383)
- Allow paper managers to submit paper revisions
- Fix error when not providing a URL for privacy policy or terms
- Use consistent order for privacy/terms links in the footer
- Fix cloning of locked events

## 8.1.33 Version 2.1.1

*Released on May 31, 2018*

## Improvements

- Add a privacy policy page linked from the footer (#1415)
- Terms & Conditions can now link to an external URL
- Show a warning to all admins if Celery is not running or outdated
- Add registration ID placeholder for badges (#3370, thanks @bpedersen2)

## Bugfixes

- Fix alignment issue in the “Indico Weeks View” timetable theme (#3367)
- Reset visibility when cloning an event to a different category (#3372)

## 8.1.34 Version 2.1

*Released on May 16, 2018*

## Major Features

- Add event roles, which are similar to local groups but within the scope of an event. They can be used both for assigning permissions within the event and also for quickly seeing which user has which role (such as “Program Committee” in the event)
- Add new *Participant Roles* (previously called *Roles*) which now shows each person’s custom event roles and whether they have registered for the event in addition to the the default roles (speaker, chairperson, etc.)
- Add visibility options to custom abstract/contribution fields so they can be restricted to be editable/visible only for event managers or authors/submitters instad of anyone who can see the abstract/contribution
- Provide new interface to import registrations/contributions from a CSV file (#3144)
- Rework how access/permissions are managed. Now all access and management privileges can be assigned from a single place on the protection management page.

## Improvements

- Allow specifying a default session for a track which will then be used by default when accepting an abstract in that track (#3069)
- Allow marking contribution types as private so they cannot be selected by users submitting an abstract (#3138)
- Add support for boolean (yes/no) and freetext questions in abstract reviewing (#3175)
- Support event cloning with monthly recurrence on the last day of the month (#1580)
- Add support for custom session types (#3189)
- Move poster session flag from session settings to session type settings
- Add contribution cloning within an event (#3207)
- Add option to include the event description in reminder emails (#3157, thanks @bpedersen2)
- Pin default themes to the top for event managers (#3166)

- Add user setting whether to show future events or not by default in a category. Also keep the per-category status in the session (#3233, thanks @bpedersen2)
- Keep page titles in sync with conference menu item titles (#3236)
- Add option to hide an attachment folder in the display areas of an event (#3181, thanks @bpedersen2)
- Improve flower redirect URI generation (#3187, thanks @bpedersen2)
- When blocking a user account, the user will be forcefully logged out in addition to being prevented from logging in
- Show track-related columns in abstract list only if there are tracks defined for the event (#2813)
- Show warning box to inform that reviewer roles do not apply when an event has no tracks (#2919)
- Allow specifying min/max length for registration form text fields (#3193, thanks @bpedersen2)
- Add settings to configure the scale of ‘rating’ questions in paper reviewing
- Show a nicer error message when entering an excessively high base registration fee (#3260)
- Use proper British English for person titles (#3279)
- Add event keywords in meta tags (#3262, thanks @bpedersen2)
- Improve sorting by date fields in the registrant list
- Use the user’s preferred name format in more places
- Add “back to conference” link when viewing a conference timetable using a meeting theme (#3297, thanks @bpedersen2)
- Allow definition lists in places where Markdown or HTML is accepted (#3325)
- Include event date/time in registration emails (#3337)
- Allow div/span/pre with classes when writing raw HTML in CKEditor (#3332, thanks @bpedersen2)
- Sort abstract authors/speakers by last name (#3340)
- Improve machine-readable metadata for events and categories (#3287, thanks @bpedersen2)

## Bugfixes

- Fix selecting a person’s title in a different language than English
- Fix display issue in “now happening” (#3278)
- Fix error when displaying the value of an accommodation field in the registrant list and someone has the “no accomodation” option selected (#3272, thanks @bpedersen2)
- Use the ‘Reviewing’ realm when logging actions from the abstract/paper reviewing modules
- Fix error when printing badges/posters with empty static text fields (#3290)
- Fix error when generating a PDF timetable including contribution abstracts (#3289)
- Do not require management access to a category to select a badge template from it as a backside.
- Fix breadcrumb metadata (#3321, thanks @bpedersen2)
- Fix error when accessing certain registration pages without an active registration
- Use event timezone when displaying event log entries (#3354)
- Correctly render most markdown elements when generating a programme PDF (#3351)

- Do not send any emails when trying to approve/reject a registration that is not pending (#3358)

## Internal Changes

- Rename *Roles* in ACL entries to *Permissions*. This especially affects the `can_manage` method whose `role` argument has been renamed to `permission` (#3057)
  - Add new `registration_checkin_updated` signal that can be used by plugins to perform an action when the checkin state of a registration changes (#3161, thanks @bpedersen2)
  - Add new signals that allow plugins to run custom code at the various stages of the RH execution and replace/modify the final response (#3227)
  - Add support for building plugin wheels with date/commit-suffixed version numbers (#3232, thanks @driehle)
- 

## 8.1.35 Version 2.0.3

*Released on March 15, 2018*

### Security fixes

- Do not show contribution information (metadata including title, speakers and a partial description) in the contribution list unless the user has access to a contribution

### Improvements

- Show more suitable message when a service request is auto-accepted (#3264)

## 8.1.36 Version 2.0.2

*Released on March 07, 2018*

### Security fixes

- Update `bleach` to fix an XSS vulnerability

### Improvements

- Warn when editing a speaker/author would result in duplicate emails

### Bugfixes

- Take ‘center’ orientation of badge/poster backgrounds into account (#3238, thanks @bpedersen2)
- Fail nicely when trying to register a local account with an already-used email confirmation link (#3250)



### 8.1.37 Version 2.0.1

*Released on February 6, 2018*

#### Improvements

- Add support for admin-only designer placeholders. Such placeholders can be provided by custom plugins and only be used in the designer by Indico admins (#3210)
- Sort contribution types alphabetically
- Add folding indicators when printing foldable badges (#3216)

#### Bugfixes

- Fix LaTeX rendering issue when consecutive lines starting with [ were present (#3203)
- Do not allow managers to retrieve tickets for registrants for whom ticket access is blocked by a plugin (#3208)
- Log a warning instead of an exception if the Indico version check fails (#3209)
- Wrap long lines in event log entries instead of truncating them
- Properly show message about empty agenda in reminders that have “Include agenda” enabled but an empty timetable
- Fix overly long contribution type names pushing edit/delete buttons outside the visible area (#3215)
- Only apply plugin-imposed ticket download restrictions for tickets, not for normal badges.
- Fix switching between badge sides in IE11 (#3214)
- Do not show poster templates as possible backsides for badges
- Convert alpha-channel transparency to white in PDF backgrounds
- Make number inputs big enough to show 5 digits in chrome
- Sort chairperson list on lecture pages
- Remove whitespace before commas in speaker lists
- Hide author UI for subcontribution speakers (#3222)

### 8.1.38 Version 2.0

*Released on January 12, 2018*

#### Improvements

- Add `author_type` and `is_speaker` fields for persons in the JSON abstract export
- Add legacy redirect for `conferenceTimeTable.py`

## Bugfixes

- Fix unicode error when searching external users from the “Search Users” dialog
- Fix missing event management menu/layout when creating a material package from the event management area
- Fix error when viewing a contribution with co-authors
- Fix sorting of registration form items not working anymore after moving/disabling some items
- Fix error after updating from 2.0rc1 if there are cached Mako templates
- Fix error when retrieving an image referenced in an abstract fails
- Fix rendering of time pickers in recent Firefox versions (#3194)
- Fix error when trying to use the html serializer with the timetable API
- Fix error when receiving invalid payment events that should be ignored
- Fix last occurrence not being created when cloning events (#3192)
- Fix multiple links in the same line being replaced with the first one when converting abstracts/contributions to PDF (#2816)
- Fix PDF generation when there are links with & in the URL
- Fix incorrect spacing in abstract author/speaker lists (#3205)

## 8.1.39 Version 2.0rc2

*Released on December 8, 2017*

## Improvements

- Allow changing the reloader used by the dev server (#3150)

## Bugfixes

- Do not show borders above/below the message in registration emails unless both the header and body blocks are used (#3151)
- Roll-back the database transaction when an error occurs.
- Fix rendering of the LaTeX error box (#3163)
- Fix “N/A” being displayed in a survey result if 0 is entered in a number field
- Fix “N/A” not being displayed in a survey result if nothing is selected in a multi-choice select field
- Fix error when using `target_*` placeholders in abstract notification emails for actions other than “Merged” (#3171)
- Show full track title in tooltips on abstract pages
- Show correct review indicators when a reviewer still has to review an abstract in a different track
- Fix unicode error when searching external users in an LDAP backend

## Internal Changes

- Remove `SCSS_DEBUG_INFO` config option.

### 8.1.40 Version 2.0rc1

*Released on November 10, 2017*

## Improvements

- Hide category field in event creation dialog if there are no subcategories (#3112)
- Remove length limit from registration form field captions (#3119)
- Use semicolons instead of commas as separator when exporting list values (such as multi-select registration form fields) to CSV or Excel (#3060)
- Use custom site title in page title (#3018)
- Allow manually entering dates in datetime fields (#3136)
- Send emails through a celery task. This ensures users do not get an error if the mail server is temporarily unavailable. Sending an email is also retried for a while in case of failure. In case of a persistent failure the email is dumped to the temp directory and can be re-sent manually using the new `indico resend_email` command (#3121)
- Reject requests containing NUL bytes in the query string (#3142)

## Bugfixes

- Do not intercept HTTP exceptions containing a custom response. When raising such exceptions we do not want the default handling but rather send the custom response to the client.
- Do not apply margin for empty root category sidebar (#3116, thanks @nop33)
- Fix alignment of info-grid items on main conference page (#3126)
- Properly align the label of the attachment folder title field
- Fix some rare unicode errors during exception handling/logging
- Clarify messages in session block rescheduling dialogs (#3080)
- Fix event header bar in IE11 (#3135)
- Fix footer on login page (#3132)
- Use correct module name for abstract notification emails in the event log
- Remove linebreaks from email subject in paper review notifications
- Fix extra padding in the CFA roles dialog (#3129)
- Do not show an extra day in timetable management if an event begins before a DST change
- Disable caching when retrieving the list of unscheduled contributions
- Process placeholders in the subject when emailing registrants
- Fix Shibboleth login with non-ascii names (#3143)

## Internal Changes

- Add new `is_ticket_blocked` signal that can be used by plugins to disable ticket downloads for a registration.

### 8.1.41 Version 2.0a1

*Released on October 20, 2017*

This is the first release of the 2.0 series, which is an almost complete rewrite of Indico based on a modern software stack and PostgreSQL.

## CHAPTER 9

---

### Indices and tables

---

- `genindex`
- `modindex`



## 10.1 Contact

### 10.1.1 Website

The official website of Indico is [getindico.io](https://getindico.io), there you can find useful information related to the project.

### 10.1.2 IRC

We use IRC as our main means of real-time communication with the development community. Get in touch through the official [#indico](#) channel on Libera.Chat ([irc.libera.chat](https://irc.libera.chat)). It is also accessible through [Matrix](#).

### 10.1.3 Forum

For more elaborate questions and discussions we encourage you to use our [discussion forum](#).

### 10.1.4 Issue tracker

We use [GitHub issues](#) for specific bug reports and feature requests. Support enquiries are better suited for the IRC channel or the forums.

### 10.1.5 Twitter

Indico has an official Twitter account, [@getindico](#) which is occasionally used for announcements.





### i

`indico.core.oauth`, 304  
`indico.core.oauth.models.applications`, 304  
`indico.core.oauth.models.tokens`, 307  
`indico.core.plugins`, 77  
`indico.modules.attachments`, 281  
`indico.modules.attachments.models.attachments`, 281  
`indico.modules.attachments.models.folders`, 283  
`indico.modules.attachments.models.principals`, 285  
`indico.modules.attachments.operations`, 287  
`indico.modules.attachments.preview`, 288  
`indico.modules.attachments.util`, 287  
`indico.modules.auth`, 302  
`indico.modules.auth.models.identities`, 302  
`indico.modules.auth.models.registration_requests`, 303  
`indico.modules.auth.util`, 303  
`indico.modules.categories`, 263  
`indico.modules.categories.fields`, 333  
`indico.modules.categories.models.categories`, 263  
`indico.modules.categories.models.principals`, 267  
`indico.modules.categories.models.settings`, 268  
`indico.modules.categories.operations`, 268  
`indico.modules.categories.serialize`, 269  
`indico.modules.categories.settings`, 270  
`indico.modules.categories.util`, 268  
`indico.modules.designer`, 316  
`indico.modules.designer.models.images`, 316  
`indico.modules.designer.models.templates`, 317  
`indico.modules.designer.pdf`, 318  
`indico.modules.designer.placeholders`, 319  
`indico.modules.designer.util`, 318  
`indico.modules.events`, 121  
`indico.modules.events.abstracts`, 144  
`indico.modules.events.abstracts.fields`, 328  
`indico.modules.events.abstracts.models.abstracts`, 144  
`indico.modules.events.abstracts.models.call_for_abstracts`, 148  
`indico.modules.events.abstracts.models.comments`, 149  
`indico.modules.events.abstracts.models.email_logs`, 150  
`indico.modules.events.abstracts.models.email_templates`, 150  
`indico.modules.events.abstracts.models.fields`, 151  
`indico.modules.events.abstracts.models.files`, 152  
`indico.modules.events.abstracts.models.persons`, 152  
`indico.modules.events.abstracts.models.related_tracks`, 153  
`indico.modules.events.abstracts.models.review_requests`, 153  
`indico.modules.events.abstracts.models.review_ratings`, 154  
`indico.modules.events.abstracts.models.reviews`, 154  
`indico.modules.events.abstracts.operations`, 156  
`indico.modules.events.abstracts.placeholders`, 159  
`indico.modules.events.abstracts.settings`, 164

`indico.modules.events.abstracts.util`, 157  
`indico.modules.events.agreements`, 166  
`indico.modules.events.agreements.models.agreements`, 166  
`indico.modules.events.agreements.placeholders`, 168  
`indico.modules.events.agreements.util`, 167  
`indico.modules.events.contributions`, 168  
`indico.modules.events.contributions.fields`, 331  
`indico.modules.events.contributions.models.contributions`, 169  
`indico.modules.events.contributions.models.fields`, 172  
`indico.modules.events.contributions.models.persons`, 174  
`indico.modules.events.contributions.models.principals`, 175  
`indico.modules.events.contributions.models.references`, 176  
`indico.modules.events.contributions.models.subcontributions`, 177  
`indico.modules.events.contributions.models.types`, 178  
`indico.modules.events.contributions.operations`, 179  
`indico.modules.events.contributions.util`, 180  
`indico.modules.events.features`, 181  
`indico.modules.events.features.util`, 181  
`indico.modules.events.fields`, 327  
`indico.modules.events.layout`, 182  
`indico.modules.events.layout.models.images`, 182  
`indico.modules.events.layout.models.menu`, 183  
`indico.modules.events.layout.util`, 185  
`indico.modules.events.logs`, 186  
`indico.modules.events.logs.models.entries`, 186  
`indico.modules.events.logs.renderers`, 188  
`indico.modules.events.logs.util`, 188  
`indico.modules.events.models.events`, 121  
`indico.modules.events.models.persons`, 127  
`indico.modules.events.models.principals`, 130  
`indico.modules.events.models.references`, 131  
`indico.modules.events.models.reviews`, 133  
`indico.modules.events.models.series`, 135  
`indico.modules.events.models.settings`, 135  
`indico.modules.events.models.static_list_links`, 136  
`indico.modules.events.notes`, 190  
`indico.modules.events.notes.models.notes`, 190  
`indico.modules.events.notes.util`, 192  
`indico.modules.events.operations`, 137  
`indico.modules.events.papers`, 192  
`indico.modules.events.papers.fields`, 331  
`indico.modules.events.papers.models.call_for_papers`, 193  
`indico.modules.events.papers.models.comments`, 194  
`indico.modules.events.papers.models.competences`, 195  
`indico.modules.events.papers.models.files`, 195  
`indico.modules.events.papers.models.papers`, 196  
`indico.modules.events.papers.models.review_questions`, 197  
`indico.modules.events.papers.models.review_ratings`, 198  
`indico.modules.events.papers.models.reviews`, 198  
`indico.modules.events.papers.models.revisions`, 200  
`indico.modules.events.papers.models.templates`, 202  
`indico.modules.events.papers.models.user_contributions`, 203  
`indico.modules.events.papers.operations`, 203  
`indico.modules.events.papers.util`, 204  
`indico.modules.events.payment`, 207  
`indico.modules.events.payment.models.transactions`, 207  
`indico.modules.events.payment.plugins`, 209  
`indico.modules.events.payment.util`, 208  
`indico.modules.events.persons`, 210  
`indico.modules.events.persons.operations`, 210  
`indico.modules.events.persons.placeholders`, 210  
`indico.modules.events.registration`, 212  
`indico.modules.events.registration.models.form_fields`, 217  
`indico.modules.events.registration.models.forms`, 219  
`indico.modules.events.registration.models.invitations`, 219

[221](#) [indico.modules.events.timetable, 253](#)  
[indico.modules.events.registration.models.items, 253](#)  
[223](#) [indico.modules.events.timetable.models.breaks, 253](#)  
[indico.modules.events.registration.models.sessions, 254](#)  
[212](#) [indico.modules.events.timetable.models.entries, 254](#)  
[indico.modules.events.registration.placeholders.module.events, 256](#)  
[231](#) [indico.modules.events.timetable.operations, 256](#)  
[indico.modules.events.registration.placeholders.modules.events, 258](#)  
[229](#) [indico.modules.events.timetable.reschedule, 258](#)  
[indico.modules.events.registration.settings, 257](#)  
[232](#) [indico.modules.events.timetable.util, 257](#)  
[indico.modules.events.registration.stats, 259](#)  
[233](#) [indico.modules.events.tracks, 259](#)  
[indico.modules.events.registration.util, 260](#)  
[227](#) [indico.modules.events.tracks.models.principals, 260](#)  
[indico.modules.events.reminders, 234](#) [indico.modules.events.tracks.models.tracks, 259](#)  
[indico.modules.events.reminders.models.reminders, 261](#)  
[234](#) [indico.modules.events.tracks.operations, 261](#)  
[indico.modules.events.reminders.util, 139](#)  
[236](#) [indico.modules.events.util, 139](#)  
[indico.modules.events.requests, 236](#) [indico.modules.groups, 310](#)  
[indico.modules.events.requests.base, 237](#) [indico.modules.groups.core, 310](#)  
[indico.modules.events.requests.models.requests, 311](#)  
[236](#) [indico.modules.groups.models.groups, 310](#)  
[indico.modules.events.requests.util, 237](#) [indico.modules.groups.util, 311](#)  
[indico.modules.events.sessions, 239](#) [indico.modules.networks, 325](#)  
[indico.modules.events.sessions.fields, 325](#)  
[333](#) [indico.modules.networks.fields, 333](#)  
[indico.modules.events.sessions.models.blocks, 326](#)  
[241](#) [indico.modules.networks.models.networks, 325](#)  
[indico.modules.events.sessions.models.persons, 327](#)  
[243](#) [indico.modules.networks.util, 326](#)  
[indico.modules.events.sessions.models.principals, 289](#)  
[243](#) [indico.modules.news, 326](#)  
[indico.modules.events.sessions.models.sessions, 294](#)  
[239](#) [indico.modules.news.models.news, 326](#)  
[indico.modules.events.sessions.operations, 293](#)  
[244](#) [indico.modules.news.util, 327](#)  
[indico.modules.events.sessions.util, 244](#) [indico.modules.rb, 289](#)  
[indico.modules.events.settings, 204](#) [indico.modules.rb.models.blocked\\_rooms, 294](#)  
[indico.modules.events.static, 261](#) [indico.modules.rb.models.blocking\\_principals, 294](#)  
[indico.modules.events.static.models.static, 299](#)  
[262](#) [indico.modules.rb.models.blockings, 293](#)  
[indico.modules.events.static.util, 263](#) [indico.modules.rb.models.equipment, 295](#)  
[indico.modules.events.surveys, 245](#) [indico.modules.rb.models.locations, 295](#)  
[indico.modules.events.surveys.models.items, 296](#)  
[248](#) [indico.modules.rb.models.map\\_areas, 296](#)  
[indico.modules.events.surveys.models.submissions, 292](#)  
[250](#) [indico.modules.rb.models.photos, 296](#)  
[indico.modules.events.surveys.models.surveys, 293](#)  
[245](#) [indico.modules.rb.models.reservation\\_edit\\_logs, 299](#)  
[indico.modules.events.surveys.operations, 293](#)  
[251](#) [indico.modules.rb.models.reservation\\_occurrences, 300](#)  
[indico.modules.events.surveys.util, 252](#) [indico.modules.rb.models.reservations, 296](#)  
[indico.modules.rb.models.room\\_attributes, 292](#)  
[indico.modules.rb.models.room\\_bookable\\_hours, 293](#)  
[indico.modules.rb.models.room\\_nonbookable\\_periods, 293](#)  
[indico.modules.rb.models.rooms, 289](#)

- `indico.modules.rb.models.util`, [301](#)
- `indico.modules.rb.statistics`, [302](#)
- `indico.modules.rb.util`, [301](#)
- `indico.modules.search.base`, [70](#)
- `indico.modules.search.result_schemas`,  
[71](#)
- `indico.modules.users`, [271](#)
- `indico.modules.users.ext`, [280](#)
- `indico.modules.users.models.affiliations`,  
[276](#)
- `indico.modules.users.models.emails`, [276](#)
- `indico.modules.users.models.favorites`,  
[276](#)
- `indico.modules.users.models.settings`,  
[277](#)
- `indico.modules.users.models.suggestions`,  
[276](#)
- `indico.modules.users.models.users`, [271](#)
- `indico.modules.users.operations`, [278](#)
- `indico.modules.users.util`, [279](#)
- `indico.modules.vc`, [311](#)
- `indico.modules.vc.exceptions`, [316](#)
- `indico.modules.vc.models.vc_rooms`, [312](#)
- `indico.modules.vc.plugins`, [314](#)
- `indico.modules.vc.util`, [314](#)
- `indico.web.forms.fields`, [334](#)

## A

absolute\_download\_url (in- abstract\_updated (in module in-  
dico.modules.attachments.models.attachments.Attachment dico.core.signals.event), 84  
attribute), 281 AbstractAction (class in in-  
Abstract (class in in- dico.modules.events.abstracts.models.reviews),  
dico.modules.events.abstracts.models.abstracts), 154  
144 AbstractComment (class in in-  
abstract (indico.modules.events.abstracts.models.comments.AbstractComment dico.modules.events.abstracts.models.comments),  
attribute), 149 149  
abstract (indico.modules.events.abstracts.models.email\_logs.AbstractEmailLogEntry AbstractCommentVisibility (class in in-  
attribute), 150 dico.modules.events.abstracts.models.reviews),  
abstract (indico.modules.events.abstracts.models.files.AbstractFile 155  
attribute), 152 AbstractEmailLogEntry (class in in-  
abstract (indico.modules.events.abstracts.models.reviews.AbstractReview dico.modules.events.abstracts.models.email\_logs),  
attribute), 155 150  
abstract (indico.modules.events.contributions.models.contributions.Contribution AbstractEmailTemplate (class in in-  
attribute), 169 dico.modules.events.abstracts.models.email\_templates),  
abstract\_created (in module in- 150  
dico.core.signals.event), 84 AbstractField (class in in-  
abstract\_deleted (in module in- dico.modules.events.abstracts.fields), 328  
dico.core.signals.event), 84 AbstractFieldValue (class in in-  
abstract\_id (indico.modules.events.abstracts.models.comments.AbstractComment dico.modules.events.abstracts.models.fields),  
attribute), 149 151  
abstract\_id (indico.modules.events.abstracts.models.email\_logs.AbstractEmailLogEntry (class in in-  
attribute), 150 dico.modules.events.abstracts.models.files),  
abstract\_id (indico.modules.events.abstracts.models.fields.AbstractField 153  
attribute), 151 AbstractFieldValue  
abstract\_id (indico.modules.events.abstracts.models.files.AbstractFile AbstractIDPlaceholder (class in in-  
attribute), 152 dico.modules.events.abstracts.placeholders),  
159  
abstract\_id (indico.modules.events.abstracts.models.persons.AbstractPersonLink AbstractInvitationURLPlaceholder (class in  
attribute), 152 indico.modules.events.abstracts.placeholders),  
abstract\_id (indico.modules.events.abstracts.models.reviews.AbstractReview 160  
attribute), 155 AbstractPersonLink (class in in-  
abstract\_id (indico.modules.events.contributions.models.contributions.Contribution dico.modules.events.abstracts.models.persons),  
attribute), 169 152  
abstract\_state\_changed (in module in- AbstractPersonLinkListField (class in in-  
dico.core.signals.event), 84 dico.modules.events.abstracts.fields), 328  
abstract\_title (in- AbstractPublicState (class in in-  
dico.modules.events.abstracts.settings.BOASortField dico.modules.events.abstracts.models.abstracts),  
147

AbstractReview (class in in- attribute), 299  
     dico.modules.events.abstracts.models.reviews), 155  
 AbstractReviewingState (class in in- attribute), 330  
     dico.modules.events.abstracts.models.abstracts), 147  
 AbstractReviewQuestion (class in in- attribute), 144  
     dico.modules.events.abstracts.models.review\_questions), 153  
 AbstractReviewRating (class in in- attribute), 144  
     dico.modules.events.abstracts.models.review\_ratings), 154  
 AbstractSessionPlaceholder (class in in- attribute), 167  
     dico.modules.events.abstracts.placeholders), 160  
 AbstractState (class in in- attribute), 196  
     dico.modules.events.abstracts.models.abstracts), 147  
 AbstractTitlePlaceholder (class in in- attribute), 144  
     dico.modules.events.abstracts.placeholders), 159  
 AbstractTrackPlaceholder (class in in- attribute), 144  
     dico.modules.events.abstracts.placeholders), 160  
 AbstractURLPlaceholder (class in in- attribute), 283  
     dico.modules.events.abstracts.placeholders), 160  
 accept (indico.modules.events.abstracts.models.reviews.AbstractAction attribute), 154  
 accept (indico.modules.events.papers.models.reviews.PaperAction attribute), 198  
 accept () (indico.modules.events.agreements.models.agreements.AgreementState method), 166  
 accept () (indico.modules.events.requests.base.RequestDefinitionBase class method), 237  
 accept () (indico.modules.rb.models.reservations.Reservation attribute), 297  
 accepted (indico.modules.events.abstracts.models.abstracts.AbstractPublicState attribute), 147  
 accepted (indico.modules.events.abstracts.models.abstracts.AbstractPrivateState attribute), 147  
 accepted (indico.modules.events.agreements.models.agreements.AgreementState attribute), 166  
 accepted (indico.modules.events.agreements.models.agreements.AgreementState attribute), 167  
 accepted (indico.modules.events.papers.models.revisions.PaperRevision attribute), 202  
 accepted (indico.modules.events.registration.models.invitations.Invitation attribute), 221  
 accepted (indico.modules.events.requests.models.requests.RequestState attribute), 237  
 accepted (indico.modules.rb.models.blocked\_rooms.BlockedRoom attribute), 294  
 accepted (indico.modules.rb.models.reservations.ReservationState attribute), 283  
 accepted\_condition\_types (in- dico.modules.events.abstracts.fields.EmailRuleListField attribute), 330  
 accepted\_contrib\_type (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 144  
 accepted\_contrib\_type\_id (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 144  
 accepted\_on\_behalf (in- dico.modules.events.agreements.models.agreements.AgreementState attribute), 167  
 accepted\_revision (in- dico.modules.events.papers.models.papers.Paper attribute), 196  
 accepted\_track (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 144  
 accepted\_track\_id (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 144  
 access\_key (indico.modules.attachments.models.attachments.Attachment attribute), 281  
 access\_key (indico.modules.attachments.models.folders.AttachmentFolder attribute), 283  
 access\_key (indico.modules.categories.models.categories.Category attribute), 263  
 access\_key (indico.modules.events.contributions.models.contributions.Contribution attribute), 169  
 access\_key (indico.modules.events.models.events.Event attribute), 122  
 access\_key (indico.modules.events.sessions.models.sessions.Session attribute), 240  
 access\_key (indico.modules.events.tracks.models.tracks.Track attribute), 259  
 access\_key (indico.modules.rb.models.rooms.Room attribute), 289  
 access\_token (indico.core.oauth.models.tokens.TokenModelBase attribute), 309  
 access\_token\_hash (in- dico.core.oauth.models.tokens.OAuthToken attribute), 308  
 access\_token\_id (in- dico.core.oauth.models.tokens.TokenModelBase attribute), 309  
 AccessControlListField (class in in- dico.modules.events.registration.models.invitations.Invitation attribute), 344  
 AccommodationStats (class in in- dico.modules.events.registration.stats), 233  
 acl (indico.modules.attachments.models.attachments.Attachment attribute), 281  
 acl (indico.modules.attachments.models.folders.AttachmentFolder attribute), 283

[acl\\_entries \(indico.modules.attachments.models.attachments.Attachment attribute\), 281](#)  
[acl\\_entries \(indico.modules.attachments.models.folders.AttachmentFolder attribute\), 283](#)  
[acl\\_entries \(indico.modules.categories.models.categories.Category attribute\), 264](#)  
[acl\\_entries \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 169](#)  
[acl\\_entries \(indico.modules.events.models.events.Event attribute\), 122](#)  
[acl\\_entries \(indico.modules.events.sessions.models.sessions.Session attribute\), 240](#)  
[acl\\_entries \(indico.modules.events.tracks.models.tracks.Track attribute\), 259](#)  
[acl\\_entries \(indico.modules.rb.models.rooms.Room attribute\), 289](#)  
[acl\\_event\\_settings \(in-dico.core.plugins.IndicoPlugin attribute\), 77](#)  
[acl\\_proxy\\_class \(in-dico.modules.events.settings.EventSettingsProxy attribute\), 143, 205](#)  
[acl\\_settings \(indico.core.plugins.IndicoPlugin attribute\), 77](#)  
[acl\\_settings \(indico.modules.vc.plugins.VCPluginMixin attribute\), 314](#)  
[active \(indico.modules.search.base.IndicoSearchProvider attribute\), 70](#)  
[active\\_and\\_answered \(in-dico.modules.events.surveys.models.surveys.Survey attribute\), 247](#)  
[active\\_and\\_clean \(in-dico.modules.events.surveys.models.surveys.Survey attribute\), 247](#)  
[active\\_fields \(in-dico.modules.events.registration.models.forms.RegistrationForm attribute\), 219](#)  
[active\\_fields \(in-dico.modules.events.registration.models.items.RegistrationFormSection attribute\), 225](#)  
[active\\_registrations \(in-dico.modules.events.registration.models.forms.RegistrationForm attribute\), 219](#)  
[add\\_abstract\\_files \(\) \(in module in-dico.modules.events.abstracts.operations\), 156](#)  
[add\\_attachment\\_link \(\) \(in module in-dico.modules.attachments.operations\), 287](#)  
[add\\_edit\\_log \(\) \(in-dico.modules.rb.models.reservations.Reservation method\), 297](#)  
[add\\_file\\_date\\_column \(in-dico.modules.events.abstracts.models.files.AbstractFile attribute\), 152](#)  
[add\\_file\\_date\\_column \(in-dico.modules.events.papers.models.files.PaperFile attribute\), 195](#)  
[add\\_file\\_date\\_column \(in-dico.modules.events.papers.models.templates.PaperTemplate attribute\), 202](#)  
[add\\_file\\_date\\_column \(in-dico.modules.events.registration.models.registrations.Registration attribute\), 215](#)  
[add\\_file\\_date\\_column \(in-dico.modules.events.static.models.static.StaticSite attribute\), 262](#)  
[add\\_form\\_fields \(in module in-dico.core.signals.core\), 83](#)  
[add\\_principal \(\) \(in-dico.modules.events.settings.EventACLProxy method\), 142, 204](#)  
[add\\_survey\\_question \(\) \(in module in-dico.modules.events.surveys.operations\), 251](#)  
[add\\_survey\\_section \(\) \(in module in-dico.modules.events.surveys.operations\), 252](#)  
[add\\_survey\\_text \(\) \(in module in-dico.modules.events.surveys.operations\), 252](#)  
[add\\_url\\_rule \(\) \(in-dico.core.plugins.IndicoPluginBlueprintSetupState method\), 79](#)  
[additional\\_info \(in-dico.modules.events.models.events.Event attribute\), 122](#)  
[address \(indico.modules.events.models.persons.EventPerson attribute\), 128](#)  
[address \(indico.modules.events.models.persons.PersonLinkBase attribute\), 129](#)  
[address \(indico.modules.events.registration.models.items.PersonalDataForm attribute\), 223](#)  
[address \(in-dico.modules.search.result\\_schemas.LocationResultSchema attribute\), 73](#)  
[address \(indico.modules.users.models.users.User attribute\), 272](#)  
[adjust\\_payment\\_form\\_data \(\) \(in-dico.modules.events.payment.plugins.PaymentPluginMixin method\), 209](#)  
[advanced \(indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder attribute\), 160](#)  
[advanced \(indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder attribute\), 164](#)  
[advanced \(indico.modules.events.abstracts.placeholders.SubmitterFirstNamePlaceholder attribute\), 161](#)  
[advanced \(indico.modules.events.abstracts.placeholders.SubmitterLastNamePlaceholder attribute\), 162](#)  
[advanced \(indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder attribute\), 163](#)



attribute), 163  
 advanced (indico.modules.events.abstracts.placeholders.TargetSubmissionNamePlaceholder  
 attribute), 163  
 advanced (indico.modules.events.abstracts.placeholders.TargetSubmissionNamePlaceholder  
 attribute), 163  
 advanced (indico.modules.events.registration.placeholders.RegistrationFieldPlaceholder  
 attribute), 230  
 affiliation (indico.modules.events.models.persons.EventPerson attribute), 128  
 affiliation (indico.modules.events.models.persons.PersonLinkBase attribute), 129  
 affiliation (indico.modules.events.registration.models.invitations.RegistrationInvitation  
 attribute), 222  
 affiliation (indico.modules.events.registration.models.items.PersonalityType  
 attribute), 223  
 affiliation (indico.modules.search.result\_schemas.PersonSchema attribute), 73  
 affiliation (indico.modules.users.models.users.User attribute), 272  
 after\_commit (in module indico.core.signals.core), 83  
 after\_process (in module indico.core.signals.core), 83  
 after\_registration\_form\_clone (in module indico.core.signals.event), 84  
 AggregationSchema (class in indico.modules.search.result\_schemas), 68  
 Agreement (class in indico.modules.events.agreements.models.agreements), 166  
 AgreementLinkPlaceholder (class in indico.modules.events.agreements.placeholders), 168  
 AgreementState (class in indico.modules.events.agreements.models.agreements), 167  
 all (indico.modules.events.abstracts.settings.SubmissionRightsType attribute), 165  
 all\_emails (indico.modules.users.models.users.User attribute), 272  
 all\_files (indico.modules.attachments.models.attachments.Attachment attribute), 281  
 all\_notes (indico.modules.events.models.events.Event attribute), 122  
 all\_recipients (in indico.modules.events.reminders.models.reminders.EventReminder attribute), 234  
 allocate\_friendly\_ids() (in indico.modules.events.contributions.models.contributions.Contribution class method), 169  
 allow\_access\_key (indico.modules.events.models.events.Event attribute), 122  
 allow\_attachments (in indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstract attribute), 148  
 allow\_category\_roles (in indico.modules.events.models.principals.AttachmentFolderPrincipal attribute), 286  
 allow\_comments (in indico.modules.events.models.principals.CategoryPrincipal attribute), 267  
 allow\_contributors\_in\_comments (in indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 176  
 allow\_convener\_judgment (in indico.modules.events.models.principals.EventPrincipal attribute), 130  
 allow\_convener\_track\_change (in indico.modules.events.models.settings.EventSettingPrincipal attribute), 136  
 allow\_event\_roles (in indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 243  
 allow\_event\_roles (in indico.modules.events.tracks.models.principals.TrackPrincipal attribute), 260  
 allow\_event\_roles (in indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstract attribute), 148  
 allow\_event\_roles (in indico.modules.events.abstr



*dico.modules.events.contributions.models.principals.ContributionPrincipal* attribute), 176

*dico.modules.events.sessions.models.sessions.SessionPrincipal* attribute), 240

*allow\_event\_roles* (in- *allowed* (*indico.modules.rb.models.blockings.Blocking* attribute), 293

*dico.modules.events.models.principals.EventPrincipal* attribute), 130

*allow\_event\_roles* (in- *allowed\_always* (in- *dico.modules.events.registration.models.forms.ModificationMode* attribute), 219

*dico.modules.events.models.settings.EventSettingPrincipal* attribute), 136

*allow\_event\_roles* (in- *ALLOWED\_CONTENT\_TYPE* (in- *dico.modules.attachments.preview.ImagePreviewer* attribute), 288

*dico.modules.events.sessions.models.principals.SessionPrincipal* attribute), 243

*allow\_event\_roles* (in- *ALLOWED\_CONTENT\_TYPE* (in- *dico.modules.attachments.preview.MarkdownPreviewer* attribute), 288

*dico.modules.events.tracks.models.principals.TrackPrincipal* attribute), 260

*allow\_location\_inheritance* (in- *ALLOWED\_CONTENT\_TYPE* (in- *dico.modules.attachments.preview.PDFPreviewer* attribute), 288

*dico.modules.events.models.events.Event* attribute), 122

*allow\_networks* (in- *ALLOWED\_CONTENT\_TYPE* (in- *dico.modules.attachments.preview.Previewer* attribute), 288

*dico.modules.categories.models.principals.CategoryPrincipal* attribute), 267

*allow\_networks* (in- *ALLOWED\_CONTENT\_TYPE* (in- *dico.modules.attachments.preview.TextPreviewer* attribute), 289

*dico.modules.events.models.principals.EventPrincipal* attribute), 130

*allow\_no\_access\_contact* (in- *allowed\_link\_types* (in- *dico.modules.attachments.models.folders.AttachmentFolder* attribute), 284

*dico.modules.categories.models.categories.Category* attribute), 264

*allow\_no\_access\_contact* (in- *allowed\_link\_types* (in- *dico.modules.events.notes.models.notes.EventNote* attribute), 190

*dico.modules.events.models.events.Event* attribute), 122

*allow\_pkce\_flow* (in- *allowed\_link\_types* (in- *dico.modules.rb.models.reservations.ReservationLink* attribute), 299

*dico.core.oauth.models.applications.OAuthApplication* attribute), 304

*allow\_registration\_forms* (in- *allowed\_scopes* (in- *dico.core.oauth.models.applications.OAuthApplication* attribute), 305

*dico.modules.attachments.models.principals.AttachmentPrincipal* attribute), 286

*allow\_registration\_forms* (in- *allowed\_types\_for\_editable* (in- *dico.modules.events.contributions.models.contributions.Contribution* attribute), 169

*dico.modules.attachments.models.principals.AttachmentPrincipal* attribute), 286

*allow\_registration\_forms* (in- *allowed\_until\_approved* (in- *dico.modules.events.registration.models.forms.ModificationMode* attribute), 219

*dico.modules.events.contributions.models.principals.ContributionPrincipal* attribute), 176

*allow\_registration\_forms* (in- *allowed\_until\_payment* (in- *dico.modules.events.registration.models.forms.ModificationMode* attribute), 219

*dico.modules.events.models.principals.EventPrincipal* attribute), 130

*allow\_registration\_forms* (in- *AllowEditingType* (class in *indico.modules.events.abstracts.settings*), 164

*dico.modules.events.sessions.models.principals.SessionPrincipal* attribute), 243

*allow\_relationship\_preloading* (in- *anchor* (*indico.modules.news.models.news.NewsItem* attribute), 326

*dico.modules.categories.models.categories.Category* attribute), 264

*allow\_relationship\_preloading* (in- *announcement* (*indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstract* attribute), 148

*dico.modules.events.contributions.models.contributions.Contribution* attribute), 169

*allow\_relationship\_preloading* (in- *anonymous* (*indico.modules.events.papers.models.call\_for\_papers.CallForPapers* attribute), 193

*dico.modules.events.surveys.models.surveys.Survey* attribute), 193

attribute), 246  
 answer\_data (indico.modules.events.surveys.models.submissions.SurveySubmission), 250  
 answers (indico.modules.events.surveys.models.submissions.SurveySubmission), 251  
 api\_key (indico.modules.users.models.users.User attribute), 272  
 app\_created (in module indico.core.signals.core), 83  
 app\_user\_link (in indico.core.oauth.models.tokens.OAuthToken attribute), 308  
 app\_user\_link\_id (in indico.core.oauth.models.tokens.OAuthToken attribute), 308  
 application (indico.core.oauth.models.applications.OAuthApplication), 307  
 application (indico.core.oauth.models.tokens.OAuthToken attribute), 308  
 application\_id (in indico.core.oauth.models.applications.OAuthApplication attribute), 307  
 approve () (indico.modules.rb.models.blocked\_rooms.BlockedRoom attribute), 294  
 as\_principal (indico.modules.groups.core.GroupProxy attribute), 311  
 as\_principal (indico.modules.users.models.users.User attribute), 272  
 assignees (indico.modules.events.papers.models.call\_for\_papers.CallForPapers), 193  
 attach\_ical (indico.modules.events.registration.models.forms.RegistrationForm), 219  
 attach\_ical (indico.modules.events.reminders.models.reminders.EventReminder attribute), 234  
 Attachment (class in indico.modules.attachments.models.attachments), 281  
 attachment (indico.modules.events.agreements.models.agreements.Agreement attribute), 166  
 attachment (indico.modules.search.base.SearchTarget attribute), 70  
 attachment\_access\_override (in indico.modules.networks.models.networks.IPNetworkGroup attribute), 325  
 attachment\_accessed (in module indico.core.signals.attachments), 82  
 attachment\_created (in module indico.core.signals.attachments), 82  
 attachment\_deleted (in module indico.core.signals.attachments), 82  
 attachment\_filename (in indico.modules.events.agreements.models.agreements.Agreement attribute), 166  
 ATTACHMENT\_FOLDER\_ID\_COLUMN (in indico.modules.categories.models.categories.Categories attribute), 263  
 ATTACHMENT\_FOLDER\_ID\_COLUMN (in indico.modules.events.contributions.models.contributions.Contributions attribute), 169  
 ATTACHMENT\_FOLDER\_ID\_COLUMN (in indico.modules.events.contributions.models.subcontributions.SubContributions attribute), 177  
 ATTACHMENT\_FOLDER\_ID\_COLUMN (in indico.modules.events.models.events.Event attribute), 122  
 ATTACHMENT\_FOLDER\_ID\_COLUMN (in indico.modules.events.sessions.models.sessions.Session attribute), 240  
 attachment\_id (in indico.modules.attachments.models.attachments.AttachmentFile attribute), 282  
 attachment\_id (in indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 286  
 attachment\_id (in indico.modules.search.result\_schemas.AttachmentResultSchema attribute), 72  
 ATTACHMENT\_STORAGE (built-in variable), 55  
 attachment\_type (in indico.modules.search.result\_schemas.AttachmentResultSchema attribute), 72  
 attachment\_updated (in module indico.core.signals.attachments), 82  
 AttachmentFile (class in indico.modules.attachments.models.attachments), 282  
 AttachmentFolder (class in indico.modules.attachments.models.folders), 283  
 AttachmentFolderPrincipal (class in indico.modules.attachments.models.principals), 285  
 AttachmentPrincipal (class in indico.modules.attachments.models.principals), 286  
 AttachmentResultSchema (class in indico.modules.search.result\_schemas), 72  
 attachments (indico.modules.attachments.models.folders.AttachmentFolder attribute), 284  
 AttachmentType (class in indico.modules.attachments.models.attachments), 283  
 attr (indico.modules.events.settings.EventSettingProperty attribute), 143, 205  
 attribute (indico.modules.rb.models.room\_attributes.RoomAttribute attribute), 292  
 attribute\_id (indico.modules.rb.models.room\_attributes.RoomAttribute attribute), 292  
 attributes (indico.modules.rb.models.rooms.Room attribute), 292

attribute), 289  
 AUTH\_PROVIDERS (built-in variable), 48  
 author\_type (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 152  
 author\_type (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 174  
 author\_type (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 175  
 AUTHORS\_SPEAKERS\_DISPLAY\_ORDER\_ATTR (in-dico.modules.events.abstracts.models.abstracts.AbstractPerson attribute), 144  
 AUTHORS\_SPEAKERS\_DISPLAY\_ORDER\_ATTR (in-dico.modules.events.models.persons.AuthorsSpeakersMixin method), 166  
 attribute), 127  
 AuthorsSpeakersMixin (class in in-dico.modules.events.models.persons), 127  
 AuthorType (class in in-dico.modules.events.contributions.models.persons), 174  
 available\_equipment (in-dico.modules.rb.models.rooms.Room attribute), 289  
 avatar\_bg\_color (in-dico.modules.users.models.users.User attribute), 272  
 avatar\_url (indico.modules.events.registration.models.registration.Registration attribute), 212  
 avatar\_url (indico.modules.users.models.users.User attribute), 272  
 awaiting (indico.modules.events.abstracts.models.abstracts.AbstractPublicState attribute), 147  
**B**  
 background\_color (in-dico.modules.events.sessions.models.sessions.Session attribute), 240  
 background\_color (in-dico.modules.events.timetable.models.breaks.Break attribute), 253  
 background\_image (in-dico.modules.designer.models.templates.DesignerTemplate attribute), 317  
 background\_image\_id (in-dico.modules.designer.models.templates.DesignerTemplate attribute), 317  
 background\_position (in-dico.modules.designer.pdf.TplData attribute), 319  
 backside\_template (in-dico.modules.designer.models.templates.DesignerTemplate attribute), 317  
 backside\_template\_id (in-dico.modules.designer.models.templates.DesignerTemplate attribute), 317  
 base\_price (indico.modules.events.registration.models.forms.Registration attribute), 219  
 base\_price (indico.modules.events.registration.models.registrations.Registration attribute), 212  
 BasePrice (built-in Python class), 55  
 before\_check\_registration\_email (in module indico.modules.events.registration.models.registrations.Registration), 84  
 before\_notification\_send (in module indico.core.signals.core), 83  
 before\_process (in module indico.core.signals.rh), 89  
 belongs\_to() (indico.modules.events.agreements.models.agreements.Agreement attribute), 166  
 best\_unit (indico.web.forms.fields.TimeDeltaField attribute), 340  
 billable\_data (in-dico.modules.events.registration.models.registrations.Registration attribute), 212  
 block (indico.modules.vc.models.vc\_rooms.VCRoomLinkType attribute), 313  
 blocked\_rooms (in-dico.modules.rb.models.blockings.Blocking attribute), 293  
 blocked\_rooms (in-dico.modules.rb.models.rooms.Room attribute), 289  
 BlockedRoom (class in in-dico.modules.rb.models.blocked\_rooms), 294  
 BlockedRoomState (class in in-dico.modules.rb.models.blocked\_rooms), 294  
 Blocking (class in in-dico.modules.rb.models.blockings), 293  
 blocking\_id (indico.modules.rb.models.blocked\_rooms.BlockedRoom attribute), 294  
 blocking\_id (indico.modules.rb.models.blocking\_principals.BlockingPrincipal attribute), 295  
 BlockingPrincipal (class in in-dico.modules.rb.models.blocking\_principals), 294  
 blocks (indico.modules.events.sessions.models.sessions.Session attribute), 240  
 BOASCorrespondingAuthorType (class in in-dico.modules.events.abstracts.settings), 165  
 BOALinkFormat (class in in-dico.modules.events.abstracts.settings), 165  
 board\_number (indico.modules.events.abstracts.settings.BOASortField attribute), 165  
 board\_number (indico.modules.events.contributions.models.contributions.Contribution attribute), 169  
 BOASortField (class in in-dico.modules.events.abstracts.settings), 165  
 body (indico.modules.events.abstracts.models.email\_logs.AbstractEmailLog attribute), 165

- [attribute](#)), 150  
[body](#) ([indico.modules.events.abstracts.models.email\\_templates.AbstractEmailTemplate](#) attribute), 150  
[bookable\\_hours](#) (in [indico.modules.rb.models.rooms.Room](#) attribute), 289  
[BookableHours](#) (class in [indico.modules.rb.models.room\\_bookable\\_hours](#)), 293  
[booked\\_for\\_id](#) (in [indico.modules.rb.models.reservations.Reservation](#) attribute), 297  
[booked\\_for\\_name](#) (in [indico.modules.rb.models.reservations.Reservation](#) attribute), 297  
[booked\\_for\\_user](#) (in [indico.modules.rb.models.reservations.Reservation](#) attribute), 297  
[booking\\_created](#) (in module [indico.core.signals.rb](#)), 89  
[booking\\_deleted](#) (in module [indico.core.signals.rb](#)), 89  
[booking\\_limit\\_days](#) (in [indico.modules.rb.models.rooms.Room](#) attribute), 289  
[booking\\_modified](#) (in module [indico.core.signals.rb](#)), 89  
[booking\\_occurrence\\_state\\_changed](#) (in module [indico.core.signals.rb](#)), 89  
[booking\\_reason](#) (in [indico.modules.rb.models.reservations.Reservation](#) attribute), 297  
[booking\\_state\\_changed](#) (in module [indico.core.signals.rb](#)), 89  
[both](#) ([indico.modules.events.abstracts.models.abstracts.EditTrackModel](#) attribute), 148  
[bottom\\_right\\_latitude](#) (in [indico.modules.rb.models.map\\_areas.MapArea](#) attribute), 296  
[bottom\\_right\\_longitude](#) (in [indico.modules.rb.models.map\\_areas.MapArea](#) attribute), 296  
[Break](#) (class in [indico.modules.events.timetable.models.breaks](#)), 253  
[BREAK](#) ([indico.modules.events.timetable.models.entries.TimetableEntryType](#) attribute), 256  
[break\\_](#) ([indico.modules.events.timetable.models.entries.TimetableEntryType](#) attribute), 254  
[break\\_id](#) ([indico.modules.events.timetable.models.entries.TimetableEntryType](#) attribute), 254  
[buckets](#) ([indico.modules.search.result\\_schemas.AggregationSchema](#) attribute), 68  
[BucketSchema](#) (class in [indico.modules.search.result\\_schemas](#)), 68  
[build\\_default\\_email\\_template\(\)](#) (in module [indico.modules.events.abstracts.util](#)), 157  
[build\\_menu\\_entry\\_name\(\)](#) (in module [indico.modules.events.layout.util](#)), 186  
[build\\_note\\_api\\_data\(\)](#) (in module [indico.modules.events.notes.util](#)), 192  
[build\\_note\\_legacy\\_api\\_data\(\)](#) (in module [indico.modules.events.notes.util](#)), 192  
[build\\_registration\\_api\\_data\(\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[build\\_registrations\\_api\\_data\(\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[build\\_rooms\\_spritesheet\(\)](#) (in module [indico.modules.rb.util](#)), 301  
[build\\_user\\_search\\_query\(\)](#) (in module [indico.modules.users.util](#)), 279  
[building](#) ([indico.modules.rb.models.rooms.Room](#) attribute), 289
- ## C
- [CACHE\\_DIR](#) (built-in variable), 53  
[calculate\\_price\(\)](#) (in [indico.modules.events.registration.models.form\\_fields.RegistrationFormFields](#) method), 217  
[calculate\\_rooms\\_bookable\\_time\(\)](#) (in module [indico.modules.rb.statistics](#)), 302  
[calculate\\_rooms\\_booked\\_time\(\)](#) (in module [indico.modules.rb.statistics](#)), 302  
[calculate\\_rooms\\_occupancy\(\)](#) (in module [indico.modules.rb.statistics](#)), 302  
[call\\_for\\_proposals\\_attr](#) (in [indico.modules.events.abstracts.models.abstracts.AbstractModel](#) attribute), 144  
[call\\_for\\_proposals\\_attr](#) (in [indico.modules.events.models.reviews.ProposalMixin](#) attribute), 133  
[call\\_for\\_proposals\\_attr](#) (in [indico.modules.events.papers.models.papers.Paper](#) attribute), 196  
[CallForAbstracts](#) (class in [indico.modules.events.abstracts.models.call\\_for\\_abstracts](#)), 148  
[call\\_for\\_papers](#) (class in [indico.modules.events.papers.models.call\\_for\\_papers](#)), 195  
[can\\_accept\(\)](#) ([indico.modules.rb.models.reservations.Reservation](#) method), 297  
[can\\_access](#) (in module [indico.core.signals.acl](#)), 81  
[can\\_access\(\)](#) ([indico.modules.attachments.models.attachments.Attachment](#) method), 281  
[can\\_access\(\)](#) ([indico.modules.attachments.models.folders.AttachmentFolder](#) method), 284  
[can\\_access\(\)](#) ([indico.modules.events.abstracts.models.abstracts.AbstractModel](#) method), 144

`can_access()` (`indico.modules.events.contributions.models.subcontributions.SubContribution` method), 177  
`can_access()` (`indico.modules.events.sessions.models.blocks.SessionBlock` method), 241  
`can_access()` (`indico.modules.events.timetable.models.breaks.Break` method), 264  
`can_access()` (`indico.modules.events.tracks.models.tracks.Track` method), 259  
`can_access_judging_area()` (`indico.modules.events.papers.models.call_for_papers.CallForPapers` method), 193  
`can_access_reviewing_area()` (`indico.modules.events.papers.models.call_for_papers.CallForPapers` method), 193  
`can_be_managed()` (`indico.modules.events.requests.base.RequestDefinitionBase` class method), 238  
`can_be_modified` (`indico.modules.events.registration.models.registrations.Registration` attribute), 212  
`can_be_modified` (`indico.modules.events.requests.models.requests.Request` attribute), 236  
`can_be_modified()` (`indico.modules.events.payment.plugins.PaymentPluginMixin` method), 209  
`can_be_modified()` (`indico.modules.users.models.users.User` method), 272  
`can_be_withdrawn` (`indico.modules.events.registration.models.registrations.Registration` attribute), 212  
`can_book()` (`indico.modules.rb.models.rooms.Room` method), 289  
`can_cancel()` (`indico.modules.rb.models.reservation_occurrences.ReservationOccurrence` method), 300  
`can_cancel()` (`indico.modules.rb.models.reservations.Reservation` method), 297  
`can_change_tracks()` (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 144  
`can_comment()` (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 144  
`can_comment()` (`indico.modules.events.models.reviews.ProposalMixin` method), 133  
`can_comment()` (`indico.modules.events.papers.models.papers.Paper` method), 196  
`can_convene()` (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`can_convene()` (`indico.modules.events.models.events.Event` method), 122  
`can_create_invited_abstracts()` (`indico.modules.events.abstracts.util` module method), 157  
`can_delete()` (`indico.modules.events.tracks.models.tracks.Track` method), 259  
`can_delete()` (`indico.modules.rb.models.blockings.Blocking` method), 293  
`can_delete()` (`indico.modules.rb.models.reservations.Reservation` method), 297  
`can_delete()` (`indico.modules.rb.models.rooms.Room` method), 290  
`can_display()` (`indico.modules.events.models.events.Event` method), 122  
`can_edit()` (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`can_edit()` (`indico.modules.events.abstracts.models.comments.AbstractComment` method), 149  
`can_edit()` (`indico.modules.events.abstracts.models.reviews.AbstractReview` method), 155  
`can_edit()` (`indico.modules.events.models.reviews.ProposalCommentMixin` method), 133  
`can_edit()` (`indico.modules.events.models.reviews.ProposalReviewMixin` method), 134  
`can_edit()` (`indico.modules.events.papers.models.comments.PaperReview` method), 199  
`can_edit()` (`indico.modules.events.papers.models.reviews.PaperReview` method), 199  
`can_edit_abstracts()` (`indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` method), 148  
`can_edit_note()` (`indico.modules.events.notes.util` module method), 192  
`can_edit_note()` (`indico.modules.events.sessions.models.blocks.SessionBlock` method), 242  
`can_get_all_multipass_groups` (`indico.modules.users.models.users.User` attribute), 272  
`can_judge()` (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`can_judge()` (`indico.modules.events.papers.models.papers.Paper` method), 196  
`can_lock()` (`indico.modules.events.models.events.Event` method), 122



method), 122

can\_manage (in module `indico.core.signals.acl`), 81

can\_manage () (`indico.modules.events.contributions.models.contributions.Contribution` method), 169

can\_manage () (`indico.modules.events.contributions.models.subcontributions.SubContribution` method), 177

can\_manage () (`indico.modules.events.papers.models.papers.Paper` method), 196

can\_manage () (`indico.modules.events.sessions.models.blocks.SessionBlock` method), 242

can\_manage () (`indico.modules.rb.models.rooms.Room` method), 290

can\_manage\_attachments () (in module `indico.modules.attachments.util`), 287

can\_manage\_attachments () (`indico.modules.events.sessions.models.blocks.SessionBlock` method), 242

can\_manage\_blocks () (`indico.modules.events.sessions.models.sessions.Session` method), 240

can\_manage\_contributions () (`indico.modules.events.sessions.models.sessions.Session` method), 240

can\_manage\_sessions () (in module `indico.modules.events.sessions.util`), 244

can\_manage\_vc () (`indico.modules.vc.plugins.VCPluginMixin` method), 314

can\_manage\_vc\_room () (`indico.modules.vc.plugins.VCPluginMixin` method), 314

can\_manage\_vc\_rooms () (`indico.modules.vc.plugins.VCPluginMixin` method), 314

can\_moderate () (`indico.modules.rb.models.rooms.Room` method), 290

can\_override () (`indico.modules.rb.models.blockings.Blocking` method), 293

can\_override () (`indico.modules.rb.models.rooms.Room` method), 290

CAN\_POPULATE (`indico.modules.events.abstracts.fields.EmailRuleListField` attribute), 330

CAN\_POPULATE (`indico.modules.events.abstracts.fields.TrackRoleField` attribute), 331

CAN\_POPULATE (`indico.modules.events.papers.fields.PaperEmailSettingsField` attribute), 332

CAN\_POPULATE (`indico.web.forms.fields.IndicoLocationField` attribute), 345

CAN\_POPULATE (`indico.web.forms.fields.IndicoPalettePickerField` attribute), 339

CAN\_POPULATE (`indico.web.forms.fields.JSONField` attribute), 335

CAN\_POPULATE (`indico.web.forms.fields.OccurrencesField` attribute), 335

can\_prebook () (`indico.modules.attachments.preview.PDFPreviewer` class method), 288

can\_preview () (`indico.modules.attachments.preview.Previewer` class method), 288

can\_reject () (`indico.modules.rb.models.reservation_occurrences.ReservationOccurrence` method), 300

can\_reject () (`indico.modules.rb.models.reservations.Reservation` method), 297

can\_review () (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 145

can\_review () (`indico.modules.events.models.reviews.ProposalMixin` method), 133

can\_review () (`indico.modules.events.papers.models.papers.Paper` method), 196

can\_review\_abstracts () (`indico.modules.events.tracks.models.tracks.Track` method), 259

can\_see\_reviews () (`indico.modules.events.abstracts.models.abstracts.Abstract` method), 145

can\_submit () (`indico.modules.events.papers.models.papers.Paper` method), 196

can\_submit () (`indico.modules.events.registration.models.forms.RegistrationForm` method), 219

can\_submit () (`indico.modules.events.surveys.models.surveys.Survey` method), 246

can\_submit\_abstracts () (`indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` method), 148

can\_submit\_proceedings () (`indico.modules.events.contributions.models.contributions.Contribution` method), 170

can\_swap\_entry () (in module `indico.modules.events.timetable.operations`), 256

can\_view () (`indico.modules.attachments.models.folders.AttachmentFolder` method), 284

can\_view () (`indico.modules.events.abstracts.models.comments.AbstractComment` method), 149

can\_view () (`indico.modules.events.abstracts.models.reviews.AbstractReview` method), 155

can\_view () (`indico.modules.events.papers.models.comments.PaperReview` method), 194

can\_view () (`indico.modules.events.papers.models.reviews.PaperReview` method), 199

can\_view () (`indico.modules.events.timetable.models.entries.TimetableEntry` method), 256

method), 254  
 can\_withdraw() (in- dico.modules.events.abstracts.models.abstracts.Abstract method), 145  
 cancel (indico.modules.events.payment.models.transactions.TransactionAttribute), 208  
 cancel() (indico.modules.rb.models.reservation\_occurrences.ReservationOccurrence method), 300  
 cancel() (indico.modules.rb.models.reservations.Reservation method), 297  
 cancelled (indico.modules.events.payment.models.transactions.TransactionAttribute), 208  
 cancelled (indico.modules.rb.models.reservation\_occurrences.ReservationOccurrenceAttribute), 301  
 cancelled (indico.modules.rb.models.reservations.ReservationStateAttribute), 299  
 candidate\_contrib\_types (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 145  
 candidate\_tracks (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 145  
 capacity (indico.modules.rb.models.rooms.Room attribute), 290  
 Category (class in in- dico.modules.categories.models.categories), 263  
 category (indico.core.plugins.IndicoPlugin attribute), 77  
 category (indico.modules.attachments.models.folders.AttachmentFolderAttribute), 284  
 category (indico.modules.categories.models.settings.CategorySettingAttribute), 268  
 category (indico.modules.designer.models.templates.DesignerTemplateAttribute), 317  
 category (indico.modules.events.models.events.Event attribute), 122  
 category (indico.modules.events.notes.models.notes.EventNote attribute), 190  
 category (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 209  
 category (indico.modules.rb.models.reservations.ReservationLink attribute), 299  
 category (indico.modules.search.base.SearchTarget attribute), 70  
 category (indico.modules.users.models.suggestions.SuggestedCategoryAttribute), 277  
 category (indico.modules.vc.plugins.VCPluginMixin attribute), 314  
 category\_chain\_overlaps() (in- dico.modules.events.models.events.Event class method), 122  
 CATEGORY\_CLEANUP (built-in variable), 57  
 category\_id (indico.modules.attachments.models.folders.AttachmentFolderAttribute), 267  
 category\_id (indico.modules.categories.models.principals.CategoryPrincipalAttribute), 267  
 category\_id (indico.modules.categories.models.settings.CategorySettingAttribute), 268  
 category\_id (indico.modules.designer.models.templates.DesignerTemplateAttribute), 317  
 category\_id (indico.modules.events.models.events.Event attribute), 122  
 category\_id (indico.modules.events.notes.models.notes.EventNote attribute), 190  
 category\_id (indico.modules.rb.models.reservations.ReservationLink attribute), 299  
 category\_id (indico.modules.users.models.suggestions.SuggestedCategoryAttribute), 277  
 category\_path (in- dico.modules.search.result\_schemas.ResultSchemaBase attribute), 71  
 category\_role (in- dico.modules.attachments.models.principals.AttachmentFolderPrincipalAttribute), 286  
 category\_role (in- dico.modules.attachments.models.principals.AttachmentPrincipalAttribute), 287  
 category\_role (in- dico.modules.categories.models.principals.CategoryPrincipalAttribute), 267  
 category\_role (in- dico.modules.events.contributions.models.principals.ContributionPrincipalAttribute), 176  
 category\_role (in- dico.modules.events.models.principals.EventPrincipalAttribute), 130  
 category\_role (in- dico.modules.events.models.settings.EventSettingPrincipalAttribute), 136  
 category\_role (in- dico.modules.events.sessions.models.principals.SessionPrincipalAttribute), 243  
 category\_role (in- dico.modules.events.tracks.models.principals.TrackPrincipalAttribute), 260  
 category\_role (in- dico.modules.rb.models.blocking\_principals.BlockingPrincipalAttribute), 295  
 category\_role\_id (in- dico.modules.attachments.models.principals.AttachmentFolderPrincipalAttribute), 286  
 category\_role\_id (in- dico.modules.attachments.models.principals.AttachmentPrincipalAttribute), 287  
 category\_role\_id (in- dico.modules.categories.models.principals.CategoryPrincipalAttribute), 267

category\_role\_id (in- 290  
*dico.modules.events.contributions.models.principals.ContributionPrincipal* (in-  
 attribute), 176 *dico.modules.rb.models.rooms.Room* method),  
 category\_role\_id (in- 290  
*dico.modules.events.models.principals.EventPrincipal* check\_client () (in-  
 attribute), 130 *dico.core.oauth.models.tokens.OAuthToken*  
 category\_role\_id (in- method), 308  
*dico.modules.events.models.settings.EventSettingPrincipal* client\_secret () (in-  
 attribute), 136 *dico.core.oauth.models.applications.OAuthApplication*  
 category\_role\_id (in- method), 305  
*dico.modules.events.sessions.models.principals.SessionPrincipal* point\_auth\_method () (in-  
 attribute), 243 *dico.core.oauth.models.applications.OAuthApplication*  
 category\_role\_id (in- method), 305  
*dico.modules.events.tracks.models.principals.TrackPrincipal* event\_locked () (in module in-  
 attribute), 260 *dico.modules.events.util*), 139  
 category\_role\_id (in- check\_grant\_type () (in- (in-  
*dico.modules.rb.models.blocking\_principals.BlockingPrincipal* *dico.core.oauth.models.applications.OAuthApplication*  
 attribute), 295 method), 305  
 category\_roles (in- check\_password\_secure (in module in-  
*dico.modules.events.abstracts.fields.TrackRoleField* *dico.core.signals.core*), 83  
 attribute), 331 check\_permissions () (in module in-  
 CategoryField (class in in- *dico.modules.events.util*), 139  
*dico.modules.categories.fields*), 333 check\_redirect\_uri () (in-  
 CategoryPrincipal (class in in- *dico.core.oauth.models.applications.OAuthApplication*  
*dico.modules.categories.models.principals*), method), 305  
 267 check\_registration\_email () (in module in-  
 CategorySetting (class in in- *dico.modules.events.registration.util*), 227  
*dico.modules.categories.models.settings*), check\_response\_type () (in-  
 268 *dico.core.oauth.models.applications.OAuthApplication*  
 CategorySettingsProxy (class in in- method), 305  
*dico.modules.categories.settings*), 270 checked\_in (*indico.modules.events.registration.models.registrations.Registration*  
 CategoryTitlePlaceholder (class in in- attribute), 212  
*dico.modules.designer.placeholders*), 323 checked\_in\_dt (in-  
 CELERY\_BROKER (built-in variable), 49 *dico.modules.events.registration.models.registrations.Registration*  
 CELERY\_CONFIG (built-in variable), 49 attribute), 212  
 CELERY\_RESULT\_BACKEND (built-in variable), 49 checkin (*indico.core.oauth.models.applications.SystemAppType*  
 Cell (class in *indico.modules.events.registration.stats*), attribute), 307  
 233 children (*indico.modules.categories.models.categories.Category*  
 cfa (*indico.modules.events.models.events.Event* at- attribute), 264  
 tribute), 122 children (*indico.modules.events.layout.models.menu.MenuEntry*  
 cfp (*indico.modules.events.models.events.Event* at- attribute), 183  
 tribute), 122 children (*indico.modules.events.registration.models.form\_fields.RegistrationForm*  
 cfp (*indico.modules.events.models.reviews.ProposalMixin* attribute), 217  
 attribute), 133 children (*indico.modules.events.registration.models.form\_fields.RegistrationForm*  
 chain\_query (*indico.modules.categories.models.categories.Category* attribute), 218  
 attribute), 264 children (*indico.modules.events.registration.models.items.RegistrationForm*  
 change (*indico.modules.events.logs.models.entries.EventLogKind* attribute), 223  
 attribute), 187 children (*indico.modules.events.registration.models.items.RegistrationForm*  
 change\_tracks (in- attribute), 224  
*dico.modules.events.abstracts.models.reviews.AbstractAction* (in *indico.modules.events.registration.models.items.RegistrationForm*  
 attribute), 154 attribute), 225  
 check\_access (in module *indico.core.signals.rh*), 89 children (*indico.modules.events.registration.models.items.RegistrationForm*  
 check\_advance\_days () (in- attribute), 226  
*dico.modules.rb.models.rooms.Room* method), children (*indico.modules.events.surveys.models.items.SurveySection*



attribute), 249  
 children (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254  
 choices (indico.web.forms.fields.RelativeDeltaField attribute), 346  
 choices (indico.web.forms.fields.TimeDeltaField attribute), 340  
 clear\_boa\_cache() (in module indico.modules.events.abstracts.util), 157  
 cli (in module indico.core.signals.plugin), 88  
 client\_id (indico.core.oauth.models.applications.OAuthApplication attribute), 306  
 client\_secret (in module indico.core.oauth.models.applications.OAuthApplication attribute), 306  
 clone\_event() (in module indico.modules.events.operations), 137  
 clone\_into\_event() (in module indico.modules.events.operations), 138  
 clone\_room() (indico.modules.vc.plugins.VCPluginMixin method), 314  
 cloned (in module indico.core.signals.event), 84  
 cloned\_from (indico.modules.events.models.events.Event attribute), 122  
 cloned\_from\_id (in module indico.modules.events.models.events.Event attribute), 122  
 close() (indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstracts method), 148  
 close() (indico.modules.events.papers.models.call\_for\_papers.CallForPapers method), 193  
 close() (indico.modules.events.surveys.models.surveys.Survey method), 246  
 close\_cfa() (in module indico.modules.events.abstracts.operations), 156  
 close\_cfp() (in module indico.modules.events.papers.operations), 203  
 CoAuthorsPlaceholder (class in module indico.modules.events.abstracts.placeholders), 161  
 code (indico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 code (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 177  
 code (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 242  
 code (indico.modules.events.sessions.models.sessions.Session attribute), 240  
 code (indico.modules.events.tracks.models.tracks.Track attribute), 259  
 collect\_static\_files() (in module indico.modules.events.static.util), 263  
 colorlinks (indico.modules.events.abstracts.settings.BOALinkFormat attribute), 165  
 column (indico.modules.events.registration.models.items.PersonalDataTypes attribute), 223  
 comment (indico.modules.auth.models.registration\_requests.RegistrationRequest attribute), 303  
 comment (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 155  
 comment (indico.modules.events.papers.models.reviews.PaperReview attribute), 199  
 comment (indico.modules.events.requests.models.requests.Request attribute), 236  
 comments (indico.modules.rb.models.rooms.Room attribute), 290  
 COMMUNITY\_HUB\_URL (built-in variable), 52  
 competences (indico.modules.events.papers.models.competences.PaperCompetence attribute), 195  
 complete (indico.modules.events.payment.models.transactions.Transaction attribute), 208  
 complete (indico.modules.events.registration.models.registrations.Registration attribute), 217  
 condition\_choices (in module indico.modules.events.abstracts.fields.EmailRuleListField attribute), 330  
 condition\_class\_map (in module indico.modules.events.abstracts.fields.EmailRuleListField attribute), 330  
 Conflicting (indico.modules.events.abstracts.models.abstracts.AbstractConflicting attribute), 147  
 ConflictingOccurrences, 296  
 contact\_email (in module indico.modules.rb.models.reservations.Reservation attribute), 297  
 contact\_emails (in module indico.modules.events.models.events.Event attribute), 122  
 contact\_info (indico.modules.events.registration.models.forms.RegistrationForm attribute), 219  
 contains\_contributions (in module indico.modules.events.models.events.Event attribute), 122  
 contains\_ip() (in module indico.modules.networks.models.networks.IPNetworkGroup method), 325  
 contains\_user() (in module indico.modules.events.settings.EventACLProxy method), 142, 204

content	(indico.modules.events.papers.models.reviews.PaperReviewType attribute), 200	content	(indico.modules.vc.models.vc_rooms.VCRoomLinkType attribute), 313
content	(indico.modules.news.models.news.NewsItem attribute), 326	contribution_count	(indico.modules.events.sessions.models.blocks.SessionBlock attribute), 242
content	(indico.modules.search.result_schemas.EventNoteResultSchema attribute), 73	contribution_created	(in module indico.core.signals.event), 84
content	(indico.modules.search.result_schemas.HighlightSchema attribute), 73	contribution_deleted	(in module indico.core.signals.event), 84
content_review_questions	(indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 193	contribution_field	(indico.modules.events.abstracts.models.fields.AbstractFieldValue attribute), 151
content_reviewer_deadline	(indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 193	contribution_field	(indico.modules.events.contributions.models.fields.ContributionField attribute), 173
content_reviewer_deadline_enforced	(indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 193	contribution_field	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_reviewers	(indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 193	contribution_field_backref_name	(indico.modules.events.abstracts.models.fields.AbstractFieldValue attribute), 152
content_reviewing_enabled	(indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 193	contribution_field_backref_name	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.attachments.models.attachments.AttachmentType attribute), 283	contribution_field_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.designer.models.images.DesignerImageFile attribute), 317	contribution_id	(indico.modules.events.static.models.static.StaticSite attribute), 284
content_type	(indico.modules.events.abstracts.models.files.AbstractFile attribute), 152	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.events.layout.models.images.ImageFile attribute), 182	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.events.papers.models.files.PaperFile attribute), 195	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.events.papers.models.templates.PaperTemplate attribute), 202	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.events.registration.models.registration.RegistrationData attribute), 215	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
content_type	(indico.modules.events.static.models.static.StaticSite attribute), 262	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
Contribution	(class in indico.modules.events.contributions.models.contributions), 169	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
contribution	(indico.modules.attachments.models.folders.AttachmentFolder attribute), 284	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
contribution	(indico.modules.events.notes.models.notes.EventNote attribute), 190	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
contribution	(indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
CONTRIBUTION	(indico.modules.events.timetable.models.entries.TimetableEntry attribute), 256	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
contribution	(indico.modules.rb.models.reservations.ReservationLink attribute), 299	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174
contribution	(indico.modules.search.base.SearchTarget attribute), 70	contribution_id	(indico.modules.events.contributions.models.fields.ContributionField attribute), 174

`contribution_id` (in- `ContributionsPlaceholder` (class in `indico.modules.events.notes.models.notes.EventNote` `dico.modules.events.persons.placeholders`), attribute), 190 210  
`contribution_id` (in- `ContributionType` (class in `indico.modules.events.timetable.models.entries.TimetableEntry` `dico.modules.events.contributions.models.types`), attribute), 254 178  
`contribution_id` (in- `ContributionTypePlaceholder` (class in `indico.modules.rb.models.reservations.ReservationLink` `dico.modules.events.abstracts.placeholders`), attribute), 299 164  
`contribution_id` (in- `ContributionURLPlaceholder` (class in `indico.modules.search.result_schemas.AttachmentResultSchema` `dico.modules.events.abstracts.placeholders`), attribute), 72 164  
`contribution_id` (in- `contributors` (`indico.modules.events.abstracts.models.reviews.AbstractContribution` attribute), 155  
`dico.modules.search.result_schemas.ContributionResultSchema` attribute), 71 `contributors` (`indico.modules.events.papers.models.reviews.PaperComment` attribute), 199  
`contribution_id` (in- attribute), 199  
`dico.modules.search.result_schemas.EventNoteResultSchema` attribute), 73 `contributors` (`indico.modules.events.abstracts.models.reviews.AbstractContribution` attribute), 155  
`contribution_id` (in- `conveners` (`indico.modules.events.sessions.models.sessions.Session` attribute), 240  
`dico.modules.vc.models.vc_rooms.VCRoomEventAssociation` attribute), 312 `count` (`indico.modules.search.result_schemas.BucketSchema` attribute), 68  
`contribution_submitters` (in- attribute), 68  
`dico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` attribute), 148 `create_abstract` (`indico.modules.events.registration.models.items.PersonalDataForm` attribute), 223  
`contribution_type_row()` (in module `indico.modules.events.contributions.util`), 180 `create()` (`indico.modules.events.models.static_list_links.StaticListLink` class method), 137  
`contribution_updated` (in module `indico.core.signals.event`), 84 `create_abstract()` (in module `indico.modules.events.abstracts.operations`), 156  
`ContributionField` (class in `indico.modules.events.contributions.models.fields`), 172 `create_abstract_comment()` (in module `indico.modules.events.abstracts.operations`), 156  
`ContributionFieldValue` (class in `indico.modules.events.contributions.models.fields`), 173 `create_abstract_review()` (in module `indico.modules.events.abstracts.operations`), 156  
`ContributionFieldValueBase` (class in `indico.modules.events.contributions.models.fields`), 174 `create_boa()` (in module `indico.modules.events.abstracts.util`), 157  
`ContributionFieldVisibility` (class in `indico.modules.events.contributions.models.fields`), 174 `create_boa_tex()` (in module `indico.modules.events.abstracts.util`), 157  
`ContributionPersonLink` (class in `indico.modules.events.contributions.models.persons`), 174 `create_break_entry()` (in module `indico.modules.events.timetable.operations`), 256  
`ContributionPersonLinkListField` (class in `indico.modules.events.contributions.fields`), 331 `create_category()` (in module `indico.modules.categories.operations`), 268  
`ContributionPrincipal` (class in `indico.modules.events.contributions.models.principals`), 175 `create_comment()` (in module `indico.modules.events.papers.operations`), 203  
`ContributionReference` (class in `indico.modules.events.contributions.models.references`), 176 `create_comment_endpoint` (in- `dico.modules.events.abstracts.models.abstracts.AbstractContribution` attribute), 145  
`ContributionResultSchema` (class in `indico.modules.search.result_schemas`), 71 `create_comment_endpoint` (in- `dico.modules.events.models.reviews.ProposalMixin` attribute), 133  
`create_competences()` (in module `indico.modules.events.models.reviews.ProposalMixin` attribute), 133

[dico.modules.events.papers.operations](#)),  
[203](#)  
[create\\_contribution\(\)](#) (in module [indico.modules.events.contributions.operations](#)),  
[179](#)  
[create\\_contribution\\_from\\_abstract\(\)](#)  
(in module [indico.modules.events.contributions.operations](#)),  
[179](#)  
[create\\_event\(\)](#) (in module [indico.modules.events.operations](#)), [138](#)  
[create\\_event\\_label\(\)](#) (in module [indico.modules.events.operations](#)), [138](#)  
[create\\_event\\_logo\\_tmp\\_file\(\)](#) (in module [indico.modules.events.util](#)), [139](#)  
[create\\_event\\_references\(\)](#) (in module [indico.modules.events.operations](#)), [138](#)  
[create\\_form\(\)](#) (in module [indico.modules.events.requests.base.RequestDefinitionBase](#)  
class method), [238](#)  
[create\\_form\(\)](#) (in module [indico.modules.vc.plugins.VCPluginMixin](#)  
method), [314](#)  
[create\\_from\\_data\(\)](#) (in module [indico.modules.events.agreements.models.agreements.Agreement](#)  
static method), [166](#)  
[create\\_from\\_data\(\)](#) (in module [indico.modules.rb.models.reservations.Reservation](#)  
class method), [297](#)  
[create\\_from\\_email\(\)](#) (in module [indico.modules.events.abstracts.models.email\\_logs.AbstractEmailLogEntry](#)  
class method), [150](#)  
[create\\_from\\_user\(\)](#) (in module [indico.modules.events.models.persons.EventPerson](#)  
class method), [128](#)  
[create\\_judgment\\_endpoint\(\)](#) (in module [indico.modules.events.abstracts.models.abstracts.Abstract](#)  
attribute), [145](#)  
[create\\_judgment\\_endpoint\(\)](#) (in module [indico.modules.events.models.reviews.ProposalMixin](#)  
attribute), [133](#)  
[create\\_manager\\_form\(\)](#) (in module [indico.modules.events.requests.base.RequestDefinitionBase](#)  
class method), [238](#)  
[create\\_mock\\_abstract\(\)](#) (in module [indico.modules.events.abstracts.util](#)), [157](#)  
[create\\_next\(\)](#) (in module [indico.modules.events.payment.models.transactions.PaymentTransaction](#)  
class method), [207](#)  
[create\\_occurrences\(\)](#) (in module [indico.modules.rb.models.reservations.Reservation](#)  
method), [297](#)  
[create\\_paper\\_revision\(\)](#) (in module [indico.modules.events.papers.operations](#)), [203](#)  
[create\\_paper\\_template\(\)](#) (in module [indico.modules.events.papers.operations](#)), [203](#)  
[create\\_personal\\_data\\_fields\(\)](#) (in module [indico.modules.events.registration.util](#)), [227](#)  
[create\\_reference\\_type\(\)](#) (in module [indico.modules.events.operations](#)), [138](#)  
[create\\_registration\(\)](#) (in module [indico.modules.events.registration.util](#)), [227](#)  
[create\\_review\(\)](#) (in module [indico.modules.events.papers.operations](#)),  
[203](#)  
[create\\_review\\_endpoint\(\)](#) (in module [indico.modules.events.abstracts.models.abstracts.Abstract](#)  
attribute), [145](#)  
[create\\_review\\_endpoint\(\)](#) (in module [indico.modules.events.models.reviews.ProposalMixin](#)  
attribute), [133](#)  
[create\\_reviewing\\_question\(\)](#) (in module [indico.modules.events.operations](#)), [138](#)  
[create\\_revision\(\)](#) (in module [indico.modules.events.notes.models.notes.EventNote](#)  
method), [190](#)  
[create\\_room\(\)](#) (in module [indico.modules.vc.plugins.VCPluginMixin](#)  
method), [315](#)  
[create\\_series\(\)](#) (in module [indico.modules.rb.models.reservation\\_occurrences.ReservationOccurrence](#)  
class method), [300](#)  
[create\\_series\\_for\\_reservation\(\)](#) (in module [indico.modules.rb.models.reservation\\_occurrences.ReservationOccurrence](#)  
class method), [300](#)  
[create\\_session\(\)](#) (in module [indico.modules.events.sessions.operations](#)),  
[244](#)  
[create\\_session\\_block\(\)](#) (in module [indico.modules.events.sessions.operations](#)),  
[244](#)  
[create\\_session\\_block\\_entry\(\)](#) (in module [indico.modules.events.timetable.operations](#)), [256](#)  
[create\\_subcontribution\(\)](#) (in module [indico.modules.events.contributions.operations](#)),  
[179](#)  
[create\\_timetable\\_entry\(\)](#) (in module [indico.modules.events.timetable.operations](#)),  
[256](#)  
[create\\_track\(\)](#) (in module [indico.modules.events.tracks.operations](#)), [261](#)  
[create\\_track\\_group\(\)](#) (in module [indico.modules.events.tracks.operations](#)), [261](#)  
[create\\_untrusted\\_persons\(\)](#) (in module [indico.modules.events.abstracts.fields.AbstractPersonLinkListField](#)  
attribute), [329](#)  
[create\\_untrusted\\_persons\(\)](#) (in module [indico.modules.events.fields.EventPersonListField](#)  
attribute), [329](#)

attribute), 327  
 create\_user() (in module indico.modules.users.operations), 278  
 created (in module indico.core.signals.category), 82  
 created (in module indico.core.signals.event), 84  
 created (indico.modules.vc.models.vc\_rooms.VCRoomStatus), 314  
 created\_by\_id (in module indico.modules.events.requests.models.requests.Request), 236  
 created\_by\_id (in module indico.modules.rb.models.blockings.Blocking), 293  
 created\_by\_id (in module indico.modules.rb.models.reservations.Reservation), 297  
 created\_by\_id (in module indico.modules.vc.models.vc\_rooms.VCRoomStatus), 312  
 created\_by\_user (in module indico.modules.events.requests.models.requests.Request), 236  
 created\_by\_user (in module indico.modules.rb.models.blockings.Blocking), 293  
 created\_by\_user (in module indico.modules.rb.models.reservations.Reservation), 297  
 created\_by\_user (in module indico.modules.vc.models.vc\_rooms.VCRoomStatus), 312  
 created\_dt (indico.core.oauth.models.tokens.OAuthToken), 309  
 created\_dt (indico.core.oauth.models.tokens.TokenModelBase), 309  
 created\_dt (indico.modules.attachments.models.attachments.AttachmentsAttachmentFile), 283  
 created\_dt (indico.modules.designer.models.images.DesignerImageFile), 317  
 created\_dt (indico.modules.events.abstracts.models.comments.AbstractComment), 149  
 created\_dt (indico.modules.events.abstracts.models.files.AbstractFile), 152  
 created\_dt (indico.modules.events.abstracts.models.reviews.AbstractReview), 155  
 created\_dt (indico.modules.events.layout.models.images.LayoutImageFile), 182  
 created\_dt (indico.modules.events.models.events.Event), 122  
 created\_dt (indico.modules.events.models.static\_list\_links.StaticListLink), 137  
 created\_dt (indico.modules.events.notes.models.notes.EventNoteRevision), 192  
 created\_dt (indico.modules.events.papers.models.comments.PaperReviewComment), 194  
 created\_dt (indico.modules.events.papers.models.files.PaperFile), 195  
 created\_dt (indico.modules.events.papers.models.reviews.PaperJudgment), 199  
 created\_dt (indico.modules.events.papers.models.reviews.PaperReview), 199  
 created\_dt (indico.modules.events.papers.models.templates.PaperTemplate), 202  
 created\_dt (indico.modules.events.registration.models.registrations.RegistrationForm), 215  
 created\_dt (indico.modules.events.reminders.models.reminders.EventReminder), 234  
 created\_dt (indico.modules.events.requests.models.requests.Request), 236  
 created\_dt (indico.modules.events.static.models.static.StaticSite), 262  
 created\_dt (indico.modules.news.models.news.NewsItem), 326  
 created\_dt (indico.modules.rb.models.blockings.Blocking), 294  
 created\_dt (indico.modules.rb.models.reservations.Reservation), 297  
 created\_dt (indico.modules.vc.models.vc\_rooms.VCRoomStatus), 312  
 creator (indico.modules.events.models.events.Event), 122  
 creator (indico.modules.events.reminders.models.reminders.EventReminder), 235  
 creator (indico.modules.events.static.models.static.StaticSite), 262  
 creator\_id (indico.modules.events.models.events.Event), 122  
 creator\_id (indico.modules.events.reminders.models.reminders.EventReminder), 235  
 creator\_id (indico.modules.events.static.models.static.StaticSite), 262  
 creator\_id (indico.modules.events.payment.models.transactions.PaymentTransaction), 207  
 creator\_id (indico.modules.events.registration.models.forms.RegistrationForm), 219  
 creator\_id (indico.modules.events.registration.models.registrations.RegistrationForm), 212  
 creator\_id (indico.modules.events.registration.models.form\_fields.RegistrationFormField), 217  
 current\_data (indico.modules.events.registration.models.items.RegistrationFormItem), 223  
 current\_data (indico.modules.events.registration.models.items.RegistrationFormItem), 225  
 current\_data (indico.modules.events.registration.models.items.RegistrationFormItem), 225  
 current\_data (indico.modules.events.registration.models.items.RegistrationFormItem), 225



[attribute](#)), 226  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.contributions.models.fields.ContributionField attribute\)](#), 174  
[dico.modules.events.registration.models.form\\_fields.RegistrationFormField](#) [attribute](#)), 174  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.logs.models.entries.EventLogEntry attribute\)](#), 187  
[dico.modules.events.registration.models.form\\_fields.RegistrationFormField](#) [attribute](#)), 218  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.models.static\\_list\\_links.StaticListLink attribute\)](#), 137  
[dico.modules.events.registration.models.items.RegistrationFormItem](#) [attribute](#)), 207  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 211  
[dico.modules.events.registration.models.items.RegistrationFormItem](#) [attribute](#)), 225  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 218  
[dico.modules.events.registration.models.items.RegistrationFormItem](#) [attribute](#)), 225  
[current\\_data\\_id](#) (in- [data \(indico.modules.events.registration.models.items.RegistrationFormItem attribute\)](#), 225  
[dico.modules.events.registration.models.items.RegistrationFormSection](#) [attribute](#)), 226  
[current\\_revision](#) (in- [data \(indico.modules.events.registration.models.items.RegistrationFormSection attribute\)](#), 225  
[dico.modules.events.notes.models.notes.EventNote](#) [attribute](#)), 190  
[current\\_revision\\_id](#) (in- [data \(indico.modules.events.registration.models.registrations.RegistrationFormSection attribute\)](#), 212  
[dico.modules.events.notes.models.notes.EventNote](#) [attribute](#)), 190  
[custom](#) ([indico.modules.users.models.users.ProfilePictureSource](#) [attribute](#)), 272  
[custom\\_boa](#) ([indico.modules.events.models.events.Event](#) [attribute](#)), 122  
[custom\\_boa\\_id](#) (in- [data \(indico.modules.events.surveys.models.submissions.SurveyAnswer attribute\)](#), 250  
[dico.modules.events.models.events.Event](#) [attribute](#)), 123  
[CUSTOM\\_COUNTRIES](#) ([built-in variable](#)), 51  
[CUSTOM\\_LANGUAGES](#) ([built-in variable](#)), 51  
[CustomFieldsMixin](#) ([class in in- \[data \\(indico.modules.vc.models.vc\\\_rooms.VCRoomEventAssociation attribute\\)\]\(#\), 312](#)  
[dico.modules.events.contributions.models.contributions](#)), 172  
[CUSTOMIZATION\\_DEBUG](#) ([built-in variable](#)), 50  
[CUSTOMIZATION\\_DIR](#) ([built-in variable](#)), 50

## D

[danger](#) ([indico.modules.categories.models.categories.EventMessageMode attribute\)](#), 266  
[data](#) ([indico.modules.auth.models.identities.Identity attribute\)](#), 302  
[data](#) ([indico.modules.designer.models.templates.DesignerTemplate attribute\)](#), 317  
[data](#) ([indico.modules.events.abstracts.models.email\\_logs.AbstractEmailLogEntry attribute\)](#), 150  
[data](#) ([indico.modules.events.abstracts.models.fields.AbstractFieldValue attribute\)](#), 152  
[data](#) ([indico.modules.events.agreements.models.agreements.Agreement attribute\)](#), 166  
[data](#) ([indico.modules.events.contributions.models.fields.ContributionField attribute\)](#), 174  
[data](#) ([indico.modules.events.logs.models.entries.EventLogEntry attribute\)](#), 187  
[data](#) ([indico.modules.events.models.static\\_list\\_links.StaticListLink attribute\)](#), 137  
[data](#) ([indico.modules.events.payment.models.transactions.PaymentTransaction attribute\)](#), 207  
[data](#) ([indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 211  
[data](#) ([indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 218  
[data](#) ([indico.modules.events.registration.models.items.RegistrationFormItem attribute\)](#), 225  
[data](#) ([indico.modules.events.registration.models.items.RegistrationFormSection attribute\)](#), 226  
[data](#) ([indico.modules.events.registration.models.items.RegistrationFormSection attribute\)](#), 225  
[data](#) ([indico.modules.events.registration.models.registrations.RegistrationFormSection attribute\)](#), 212  
[data](#) ([indico.modules.events.registration.models.registrations.RegistrationFormSection attribute\)](#), 215  
[data](#) ([indico.modules.events.requests.models.requests.Request attribute\)](#), 236  
[data](#) ([indico.modules.events.surveys.models.submissions.SurveyAnswer attribute\)](#), 250  
[data](#) ([indico.modules.rb.models.photos.Photo attribute\)](#), 296  
[data](#) ([indico.modules.vc.models.vc\\_rooms.VCRoom attribute\)](#), 312  
[data](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation attribute\)](#), 312  
[data](#) ([indico.web.forms.fields.IndicoQuerySelectMultipleField attribute\)](#), 344  
[data\\_by\\_field](#) (in- [data \(indico.modules.events.abstracts.models.abstracts.AbstractField attribute\)](#), 145  
[data\\_by\\_field](#) (in- [data \(indico.modules.events.registration.models.registrations.RegistrationFormSection attribute\)](#), 212  
[data\\_versions](#) (in- [data \(indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 217  
[data\\_versions](#) (in- [data \(indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\)](#), 218  
[data\\_versions](#) (in- [data \(indico.modules.events.registration.models.items.RegistrationFormItem attribute\)](#), 223  
[data\\_versions](#) (in-

[dico.modules.events.registration.models.items.RegistrationFormPersonDataSectionable](#)), 56  
[attribute](#)), 225      [default\\_options](#) ([indico.web.forms.fields.FileField](#) (in- [attribute](#)), 342  
[data\\_versions](#) (in- [dico.modules.events.registration.models.items.RegistrationFormSectionable](#) (in- [attribute](#)), 123  
[attribute](#)), 225      [default\\_page\\_id](#) (in- [dico.modules.events.models.events.Event](#) (in- [attribute](#)), 123  
[data\\_versions](#) (in- [dico.modules.events.registration.models.items.RegistrationFormText](#) (in- [attribute](#)), 123  
[attribute](#)), 226      [default\\_protection\\_mode](#) (in- [dico.modules.rb.models.rooms.Room](#) (in- [attribute](#)),  
[DataItem](#) (class in [dico.modules.events.registration.stats](#)), 233      [default\\_redirect\\_uri](#) (in- [dico.core.oauth.models.applications.OAuthApplication](#) (in- [attribute](#)), 306  
[date](#) ([indico.modules.rb.models.reservation\\_occurrences.ReservationOccurrence](#) (in- [attribute](#)), 300  
[DAY](#) ([indico.modules.rb.models.reservations.RepeatFrequency](#) (in- [attribute](#)), 296      [default\\_render\\_mode](#) (in- [dico.modules.categories.models.categories.Category](#) (in- [attribute](#)), 264  
[day\\_number\\_data](#) (in- [dico.web.forms.fields.IndicoWeekDayRepetitionField](#) (in- [attribute](#)), 346  
[DB\\_LOG](#) (built-in variable), 52      [default\\_render\\_mode](#) (in- [dico.modules.events.abstracts.models.abstracts.Abstract](#) (in- [attribute](#)), 145  
[db\\_schema\\_created](#) (in module [dico.core.signals.core](#)), 83      [default\\_render\\_mode](#) (in- [dico.modules.events.abstracts.models.abstracts.Abstract](#) (in- [attribute](#)), 155  
[DEBUG](#) (built-in variable), 52      [declined](#) ([indico.modules.events.registration.models.invitations.InvitationStatus](#) (in- [attribute](#)), 221  
[deep\\_children\\_query](#) (in- [dico.modules.categories.models.categories.Category](#) (in- [attribute](#)), 264      [default\\_render\\_mode](#) (in- [dico.modules.events.contributions.models.contributions.Contribution](#) (in- [attribute](#)), 170  
[default\\_badge\\_template](#) (in- [dico.modules.categories.models.categories.Category](#) (in- [attribute](#)), 264      [default\\_render\\_mode](#) (in- [dico.modules.events.contributions.models.subcontributions.SubContribution](#) (in- [attribute](#)), 177  
[default\\_badge\\_template\\_id](#) (in- [dico.modules.categories.models.categories.Category](#) (in- [attribute](#)), 264      [default\\_render\\_mode](#) (in- [dico.modules.events.models.events.Event](#) (in- [attribute](#)), 123  
[default\\_colors](#) (in- [dico.modules.events.sessions.models.sessions.Session](#) (in- [attribute](#)), 240      [default\\_render\\_mode](#) (in- [dico.modules.events.papers.models.reviews.PaperReview](#) (in- [attribute](#)), 199  
[default\\_colors](#) (in- [dico.modules.events.timetable.models.breaks.Break](#) (in- [attribute](#)), 253      [default\\_render\\_mode](#) (in- [dico.modules.events.papers.models.revisions.PaperRevision](#) (in- [attribute](#)), 201  
[default\\_contribution\\_duration](#) (in- [dico.modules.events.sessions.models.sessions.Session](#) (in- [attribute](#)), 240      [default\\_render\\_mode](#) (in- [dico.modules.events.sessions.models.sessions.Session](#) (in- [attribute](#)), 240  
[default\\_data](#) ([indico.core.oauth.models.applications.SystemAppType](#) (in- [attribute](#)), 307      [default\\_render\\_mode](#) (in- [dico.modules.events.surveys.models.items.SurveyItem](#) (in- [attribute](#)), 248  
[default\\_data](#) ([indico.modules.events.layout.models.menu.MenuEntryMixIn](#) (in- [attribute](#)), 184      [default\\_render\\_mode](#) (in- [dico.modules.events.timetable.models.breaks.Break](#) (in- [attribute](#)), 253  
[default\\_event\\_settings](#) (in- [dico.core.plugins.IndicoPlugin](#) (in- [attribute](#)), 77      [default\\_render\\_mode](#) (in- [dico.modules.events.tracks.models.tracks.Track](#) (in- [attribute](#)), 259  
[default\\_event\\_themes](#) (in- [dico.modules.categories.models.categories.Category](#) (in- [attribute](#)), 264      [default\\_session](#) (in- [dico.modules.events.tracks.models.tracks.Track](#) (in- [attribute](#)), 259  
[default\\_list\\_config](#) (in- [dico.modules.events.util.ListGeneratorBase](#) (in- [attribute](#)), 139      [default\\_session\\_id](#) (in-

`dico.modules.events.tracks.models.tracks.Track` `delete_all()` (`indico.modules.categories.settings.CategorySettingsProxy` attribute), 259

`default_settings` (`indico.core.plugins.IndicoPlugin` attribute), 77

`default_settings` (`indico.modules.events.payment.plugins.PaymentPluginMixin` attribute), 209

`default_settings` (`indico.modules.vc.plugins.VCPluginMixin` attribute), 315

`default_sort_alpha` (`indico.modules.events.abstracts.fields.AbstractPersonLinkListField` attribute), 329

`default_sort_alpha` (`indico.modules.events.fields.PersonLinkListFieldBase` attribute), 328

`default_ticket_template` (`indico.modules.categories.models.categories.Category` attribute), 264

`default_ticket_template_id` (`indico.modules.categories.models.categories.Category` attribute), 264

`DEFAULT_TIMEZONE` (built-in variable), 57

`default_user_settings` (`indico.core.plugins.IndicoPlugin` attribute), 77

`defaults` (`indico.modules.events.settings.ThemeSettingsProxy` attribute), 144, 206

`definition` (`indico.modules.events.agreements.models.agreements.Agreement` attribute), 166

`definition` (`indico.modules.events.requests.models.requests.Request` attribute), 236

`delete()` (`indico.modules.categories.settings.CategorySettingsProxy` method), 270

`delete()` (`indico.modules.events.models.events.Event` method), 123

`delete()` (`indico.modules.events.notes.models.notes.EventNote` method), 190

`delete()` (`indico.modules.events.settings.EventSettingsProxy` method), 143, 205

`delete()` (`indico.modules.users.models.settings.UserSettingsProxy` method), 277

`delete()` (`indico.modules.vc.models.vc_rooms.VCRoomEventAssociation` method), 312

`delete_abstract()` (in module `indico.modules.events.abstracts.operations`), 156

`delete_abstract_comment()` (in module `indico.modules.events.abstracts.operations`), 156

`delete_abstract_files()` (in module `indico.modules.events.abstracts.operations`), 156

`delete_all()` (`indico.modules.categories.settings.CategorySettingsProxy` method), 270

`delete_all()` (`indico.modules.events.settings.EventSettingsProxy` method), 143, 206

`delete_all()` (`indico.modules.users.models.settings.UserSettingsProxy` method), 278

`delete_category()` (in module `indico.modules.categories.operations`), 268

`delete_comment()` (in module `indico.modules.events.papers.operations`), 203

`delete_comment_endpoint` (in module `indico.modules.events.abstracts.models.abstracts.AbstractField` attribute), 145

`delete_comment_endpoint` (in module `indico.modules.events.models.reviews.ProposalMixin` attribute), 134

`delete_contribution()` (in module `indico.modules.events.contributions.operations`), 179

`delete_event_label()` (in module `indico.modules.events.operations`), 138

`delete_paper_template()` (in module `indico.modules.events.papers.operations`), 203

`delete_reference_type()` (in module `indico.modules.events.operations`), 138

`delete_reviewing_question()` (in module `indico.modules.events.operations`), 138

`delete_session()` (in module `indico.modules.events.sessions.operations`), 244

`delete_session_block()` (in module `indico.modules.events.sessions.operations`), 244

`delete_subcontribution()` (in module `indico.modules.events.contributions.operations`), 179

`delete_timetable_entry()` (in module `indico.modules.events.timetable.operations`), 256

`delete_track()` (in module `indico.modules.events.tracks.operations`), 261

`delete_track_group()` (in module `indico.modules.events.tracks.operations`), 261

`deleted` (in module `indico.core.signals.category`), 82

`deleted` (in module `indico.core.signals.event`), 84

`deleted` (`indico.modules.vc.models.vc_rooms.VCRoomStatus` attribute), 314

`description` (`indico.core.oauth.models.applications.OAuthApplication` attribute), 306

`description` (`indico.modules.attachments.models.attachments.Attachment` attribute), 281

`description` (`indico.modules.attachments.models.folders.AttachmentFolder` attribute), 284



description (indico.modules.designer.placeholders.CategoriesTitlePlaceholder (indico.modules.designer.placeholders.RegistrationPricePlaceholder), 323  
 attribute), 322

description (indico.modules.designer.placeholders.EventsDatesPlaceholder (indico.modules.designer.placeholders.RegistrationTicketQuantityPlaceholder), 319  
 attribute), 321

description (indico.modules.designer.placeholders.EventsDescriptionPlaceholder (indico.modules.designer.placeholders.RegistrationTitlePlaceholder), 319  
 attribute), 321

description (indico.modules.designer.placeholders.EventsLogoPlaceholder (indico.modules.events.abstracts.models.review\_questions.ReviewQuestion), 325  
 attribute), 153

description (indico.modules.designer.placeholders.EventsOrgTextPlaceholder (indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder), 320  
 attribute), 159

description (indico.modules.designer.placeholders.EventsRoomPlaceholder (indico.modules.events.abstracts.placeholders.AbstractInvitationPlaceholder), 324  
 attribute), 160

description (indico.modules.designer.placeholders.EventsSpeakersPlaceholder (indico.modules.events.abstracts.placeholders.AbstractSessionPlaceholder), 324  
 attribute), 160

description (indico.modules.designer.placeholders.EventsTitlePlaceholder (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder), 323  
 attribute), 159

description (indico.modules.designer.placeholders.EventsVenuePlaceholder (indico.modules.events.abstracts.placeholders.AbstractTrackPlaceholder), 324  
 attribute), 160

description (indico.modules.designer.placeholders.RegistrationAddressPlaceholder (indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder), 323  
 attribute), 160

description (indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder (indico.modules.events.abstracts.placeholders.CoAuthorsFormPlaceholder), 323  
 attribute), 161

description (indico.modules.designer.placeholders.RegistrationPhonePlaceholder (indico.modules.events.abstracts.placeholders.ContributionFormPlaceholder), 322  
 attribute), 164

description (indico.modules.designer.placeholders.RegistrationCountryPlaceholder (indico.modules.events.abstracts.placeholders.ContributionFormPlaceholder), 323  
 attribute), 164

description (indico.modules.designer.placeholders.RegistrationEmailPlaceholder (indico.modules.events.abstracts.placeholders.EventTitlePlaceholder), 322  
 attribute), 159

description (indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder (indico.modules.events.abstracts.placeholders.EventURLPlaceholder), 321  
 attribute), 159

description (indico.modules.designer.placeholders.RegistrationFriendlyIDPlaceholder (indico.modules.events.abstracts.placeholders.JudgmentCardPlaceholder), 322  
 attribute), 164

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.PrimaryAuthorPlaceholder), 320  
 attribute), 161

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.SubmitterFormPlaceholder), 320  
 attribute), 162

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.SubmitterLabelPlaceholder), 321  
 attribute), 162

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder), 321  
 attribute), 161

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder), 320  
 attribute), 162

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.TargetAbstractPlaceholder), 320  
 attribute), 162

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.TargetAbstractPlaceholder), 320  
 attribute), 163

description (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder (indico.modules.events.abstracts.placeholders.TargetSubmissionPlaceholder), 321  
 attribute), 163

description (indico.modules.designer.placeholders.RegistrationLastNamePlaceholder (indico.modules.events.abstracts.placeholders.TargetSubmissionPlaceholder), 321  
 attribute), 163

description (indico.modules.designer.placeholders.RegistrationPhonePlaceholder (indico.modules.events.abstracts.placeholders.TargetSubmissionPlaceholder), 323  
 attribute), 163

description (indico.modules.designer.placeholders.RegistrationPositionPlaceholder (indico.modules.events.agreements.placeholders.AgreementFormPlaceholder), 323  
 attribute), 168

description (indico.modules.events.agreements.placeholders.PersonNamePlaceholder attribute), 168	indico.modules.events.registration.placeholders.registration.PersonNamePlaceholder attribute), 231
description (indico.modules.events.contributions.models.fields.ContributionField attribute), 172	indico.modules.networks.models.networks.IPNetworkGroup attribute), 325
description (indico.modules.events.contributions.models.bases.ContributionType attribute), 179	indico.modules.search.result_schemas.ContributionResult attribute), 71
description (indico.modules.events.papers.models.reviews.questions.PaperReviewQuestion attribute), 197	indico.modules.search.result_schemas.EventResultSchema attribute), 71
description (indico.modules.events.papers.models.templates.PaperTemplate attribute), 202	indico.modules.search.result_schemas.HighlightSchema attribute), 73
description (indico.modules.events.persons.placeholders.EmailPlaceholder attribute), 210	DisEmailPlaceholderFile (class in indico.modules.designer.models.images), 316
description (indico.modules.events.persons.placeholders.EventLinkPlaceholder attribute), 211	DisEventLinkPlaceholder (class in indico.modules.designer.pdf), 318
description (indico.modules.events.persons.placeholders.EventTitlePlaceholder attribute), 211	DisEventTitlePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.persons.placeholders.FirstNamePlaceholder attribute), 211	DisFirstNamePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.persons.placeholders.LastNamePlaceholder attribute), 211	DisLastNamePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.persons.placeholders.RegisterLinkPlaceholder attribute), 212	DisRegisterLinkPlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.form_fields.registration.RegistrationFormField attribute), 217	DisRegistrationFormField (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.form_fields.registration.RegistrationFormPersonalDataField attribute), 218	DisRegistrationFormPersonalDataField (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 223	DisRegistrationFormSection (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.items.RegistrationFormPersonalDataSection attribute), 225	DisRegistrationFormPersonalDataSection (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 225	DisRegistrationFormSection (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 226	DisRegistrationFormSection (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.invitations.FirstNamePlaceholder attribute), 231	DisInvitationFirstNamePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.invitations.EventLinkPlaceholder attribute), 232	DisInvitationEventLinkPlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.invitations.EventTitlePlaceholder attribute), 232	DisInvitationEventTitlePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.EventLinkPlaceholder attribute), 229	DisRegistrationEventLinkPlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.EventTitlePlaceholder attribute), 229	DisRegistrationEventTitlePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.FirstNamePlaceholder attribute), 230	DisRegistrationFirstNamePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.IDPlaceholder attribute), 230	DisRegistrationIDPlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.LastNamePlaceholder attribute), 231	DisRegistrationLastNamePlaceholder (class in indico.modules.designer.models.templates), 318
description (indico.modules.events.registration.placeholders.registrations.RegisterLinkPlaceholder attribute), 231	DisRegistrationRegisterLinkPlaceholder (class in indico.modules.designer.models.templates), 318

attribute), 249  
 display\_as\_section (in- attribute), 147  
 dico.modules.events.surveys.models.items.SurveySection (in- attribute), 249  
 display\_as\_section (in- duplicate\_of (indico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 dico.modules.events.surveys.models.items.SurveyText attribute), 250  
 display\_full\_name (in- duplicate\_of\_id (in- dico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 dico.modules.events.registration.models.registrations.Registration attribute), 213  
 display\_full\_name (in- duration (indico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 dico.modules.users.models.users.PersonMixin duration (indico.modules.events.contributions.models.subcontributions.Subcontribution attribute), 177  
 attribute), 271  
 display\_order (in- duration (indico.modules.events.models.events.Event attribute), 123  
 dico.modules.events.abstracts.models.persons.AbstractPersonLinkBase duration (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 242  
 display\_order (in- attribute), 174  
 dico.modules.events.contributions.models.persons.Contribution (in- dico.modules.events.timetable.models.breaks.Break attribute), 253  
 display\_order (in- duration (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254  
 dico.modules.events.contributions.models.persons.Subcontribution (in- duration (indico.modules.events.timetable.reschedule.RescheduleMode attribute), 258  
 attribute), 175  
 display\_order (in- duration (indico.modules.search.result\_schemas.ContributionResultSchema attribute), 71  
 dico.modules.events.models.persons.EventPersonLinkBase duration\_display (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 attribute), 130  
 display\_order (in- duration\_poster (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 dico.modules.events.sessions.models.persons.SessionBlockPersonLinkBase duration\_poster (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 attribute), 243  
 display\_order\_key (in- E  
 dico.modules.events.models.persons.PersonLinkBase attribute), 130  
 display\_order\_key\_lastname (in- earliest\_date (in- dico.web.forms.fields.IndicoDateField attribute), 345  
 dico.modules.events.models.persons.PersonLinkBase attribute), 130  
 display\_tzinfo (in- earliest\_dt (indico.web.forms.fields.IndicoDateTimeField attribute), 340  
 dico.modules.categories.models.categories.Category edit\_comment\_endpoint (in- dico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 attribute), 264  
 display\_tzinfo (in- edit\_comment\_endpoint (in- dico.modules.events.models.reviews.ProposalMixin attribute), 134  
 dico.modules.events.models.events.Event edit\_logs (indico.modules.rb.models.reservations.Reservation attribute), 297  
 attribute), 123  
 division (indico.modules.rb.models.rooms.Room attribute), 290  
 DoublePaymentTransaction, 207  
 download\_url (indico.modules.attachments.models.attachments.Attachment attribute), 281  
 download\_url (indico.modules.designer.models.images.DesignerImageFile attribute), 317  
 dr (indico.modules.users.models.users.UserTitle attribute), 276  
 dump () (indico.modules.search.base.SearchOptions method), 70  
 duplicate (indico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 147  
 duplicate\_of (indico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 duplicate\_of\_id (in- dico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 duration (indico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 duration (indico.modules.events.contributions.models.subcontributions.Subcontribution attribute), 177  
 duration (indico.modules.events.models.events.Event attribute), 123  
 duration (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 242  
 duration (indico.modules.events.timetable.models.breaks.Break attribute), 253  
 duration (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254  
 duration (indico.modules.events.timetable.reschedule.RescheduleMode attribute), 258  
 duration (indico.modules.search.result\_schemas.ContributionResultSchema attribute), 71  
 duration\_display (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 duration\_poster (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 duration\_poster (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 170  
 earliest\_date (in- dico.web.forms.fields.IndicoDateField attribute), 345  
 earliest\_dt (indico.web.forms.fields.IndicoDateTimeField attribute), 340  
 edit\_comment\_endpoint (in- dico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145  
 edit\_comment\_endpoint (in- dico.modules.events.models.reviews.ProposalMixin attribute), 134  
 edit\_logs (indico.modules.rb.models.reservations.Reservation attribute), 297  
 edit\_review\_endpoint (in- dico.modules.events.models.reviews.ProposalMixin attribute), 134  
 edit\_track\_mode (in- dico.modules.events.abstracts.models.abstracts.AbstractPersonLinkBase attribute), 145

attribute), 145  
 editable\_types (in- EmailPlaceholder (class in in-  
     dico.modules.events.models.events.Event  
     attribute), 123  
 EditableFileField (class in in- EmailRenderer (class in in-  
     dico.web.forms.fields), 344  
     dico.modules.events.logs.renderers), 188  
 EditTrackMode (class in in- EmailRuleListField (class in in-  
     dico.modules.events.abstracts.models.abstracts),  
     dico.modules.events.abstracts.fields), 329  
 emails (indico.modules.events.logs.models.entries.EventLogRealm  
     attribute), 188  
 effective\_icon\_url (in- ENABLE\_ROOMBOOKING (built-in variable), 57  
     dico.modules.categories.models.categories.CategoryPrincipals,  
     attribute), 264  
 email (indico.modules.attachments.models.principals.AttachmentPrincipal,  
     attribute), 286  
     indico.modules.events.contributions.models.contributions.ContributionPrincipal,  
     attribute), 170  
 email (indico.modules.attachments.models.principals.AttachmentPrincipal,  
     attribute), 287  
     indico.modules.rb.models.blockings.BlockingPrincipal,  
     attribute), 294  
 email (indico.modules.auth.models.registration\_requests.RegistrationRequestPrincipal,  
     attribute), 303  
     indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstractPrincipal,  
     attribute), 148  
 email (indico.modules.categories.models.principals.CategoryPrincipal,  
     attribute), 267  
     indico.modules.events.contributions.models.contributions.ContributionPrincipal,  
     attribute), 170  
 email (indico.modules.events.contributions.models.principals.ContributionPrincipal,  
     attribute), 176  
     indico.modules.events.models.events.Event  
     attribute), 123  
 email (indico.modules.events.models.persons.EventPersonend\_dt (indico.modules.events.papers.models.call\_for\_papers.CallForPaperPrincipal,  
     attribute), 128  
     attribute), 193  
 email (indico.modules.events.models.persons.PersonLinkBase\_dt (indico.modules.events.registration.models.forms.RegistrationFormPrincipal,  
     attribute), 130  
     attribute), 219  
 email (indico.modules.events.models.principals.EventPrincipal\_dt (indico.modules.events.sessions.models.blocks.SessionBlockPrincipal,  
     attribute), 130  
     attribute), 242  
 email (indico.modules.events.models.settings.EventSettingPrincipal (indico.modules.events.sessions.models.sessions.SessionPrincipal,  
     attribute), 136  
     attribute), 240  
 email (indico.modules.events.registration.models.invitations.RegistrationInvitationPrincipal (indico.modules.events.surveys.models.surveys.SurveyPrincipal,  
     attribute), 222  
     attribute), 246  
 email (indico.modules.events.registration.models.items.PersonalDataItemType (indico.modules.events.timetable.models.breaks.BreakPrincipal,  
     attribute), 223  
     attribute), 253  
 email (indico.modules.events.registration.models.registrations.RegistrationPrincipal (indico.modules.events.timetable.models.entries.TimetableEntryPrincipal,  
     attribute), 213  
     attribute), 254  
 email (indico.modules.events.sessions.models.principals.SessionPrincipal (indico.modules.rb.models.reservation\_occurrences.ReservationOccurrencePrincipal,  
     attribute), 243  
     attribute), 300  
 email (indico.modules.events.tracks.models.principals.TrackPrincipal (indico.modules.rb.models.reservations.ReservationPrincipal,  
     attribute), 260  
     attribute), 297  
 email (indico.modules.rb.models.blocking\_principals.BlockingPrincipal (indico.modules.rb.models.room\_nonbookable\_periods.NonBookablePeriodPrincipal,  
     attribute), 295  
     attribute), 293  
 email (indico.modules.users.models.emails.UserEmail end\_dt (indico.modules.search.result\_schemas.ContributionResultSchemaPrincipal,  
     attribute), 276  
     attribute), 71  
 email (indico.modules.users.models.users.User at- end\_dt (indico.modules.search.result\_schemas.EventResultSchemaPrincipal,  
     attribute), 272  
     attribute), 71  
 email\_added (in module indico.core.signals.users), 90 end\_dt\_display (in-  
 email\_template (in- dico.modules.events.contributions.models.contributions.ContributionPrincipal), 170  
     dico.modules.events.abstracts.models.email\_logs.AbstractEmailLogEntry,  
     attribute), 150  
     end\_dt\_display (in-  
 email\_template\_id (in- dico.modules.events.models.events.Event  
     dico.modules.events.abstracts.models.email\_logs.AbstractEmailLogEntry,  
     attribute), 150  
     attribute), 123  
     end\_dt\_local (indico.modules.events.models.events.Event  
     attribute), 123  
 EmailListField (class in indico.web.forms.fields),



`end_dt_override` (in `indico.modules.events.models.events.Event` attribute), 123  
`end_dt_poster` (in `indico.modules.events.contributions.models.contributions.Contribution` attribute), 170  
`end_notification_daily` (in `indico.modules.rb.models.rooms.Room` attribute), 290  
`end_notification_monthly` (in `indico.modules.rb.models.rooms.Room` attribute), 290  
`end_notification_sent` (in `indico.modules.rb.models.reservations.Reservation` attribute), 298  
`end_notification_weekly` (in `indico.modules.rb.models.rooms.Room` attribute), 290  
`end_notifications_enabled` (in `indico.modules.rb.models.rooms.Room` attribute), 290  
`end_time` (`indico.modules.rb.models.room_bookable_hours.BookableHours` attribute), 293  
`endpoint` (`indico.modules.events.util.ListGeneratorBase` attribute), 139  
`ends_after()` (`indico.modules.events.models.events.Event` method), 123  
`enforced_data` (in `indico.core.oauth.models.applications.SystemAppType` attribute), 307  
`entry_changed` (in module `indico.core.signals.acl`), 81  
`entry_parent` (`indico.modules.events.util.ListGeneratorBase` attribute), 139  
`EquipmentType` (class in `indico.modules.rb.models.equipment`), 295  
`Event` (class in `indico.modules.events.models.events`), 121  
`event` (`indico.modules.attachments.models.folders.AttachmentsFolder` attribute), 284  
`event` (`indico.modules.designer.models.templates.DesignerTemplate` attribute), 317  
`event` (`indico.modules.events.abstracts.fields.AbstractField` attribute), 328  
`event` (`indico.modules.events.abstracts.models.abstracts.Abstract` attribute), 145  
`event` (`indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate` attribute), 150  
`event` (`indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion` attribute), 153  
`event` (`indico.modules.events.agreements.models.agreements.Agreement` attribute), 166  
`event` (`indico.modules.events.contributions.models.contributions.Contribution` attribute), 170  
`event` (`indico.modules.events.contributions.models.fields.ContributionField` attribute), 172  
`event` (`indico.modules.events.contributions.models.subcontributions.SubContribution` attribute), 177  
`event` (`indico.modules.events.contributions.models.types.ContributionType` attribute), 179  
`event` (`indico.modules.events.fields.EventPersonListField` attribute), 327  
`event` (`indico.modules.events.layout.models.images.ImageFile` attribute), 182  
`event` (`indico.modules.events.layout.models.menu.EventPage` attribute), 183  
`event` (`indico.modules.events.layout.models.menu.MenuEntry` attribute), 183  
`event` (`indico.modules.events.logs.models.entries.EventLogEntry` attribute), 187  
`event` (`indico.modules.events.logs.models.entries.EventLogRealm` attribute), 188  
`event` (`indico.modules.events.models.events.Event` attribute), 123  
`event` (`indico.modules.events.models.persons.EventPerson` attribute), 128  
`event` (`indico.modules.events.models.settings.EventSetting` attribute), 135  
`event` (`indico.modules.events.models.settings.EventSettingPrincipal` attribute), 136  
`event` (`indico.modules.events.models.settings.EventSettingsMixin` attribute), 136  
`event` (`indico.modules.events.models.static_list_links.StaticListLink` attribute), 137  
`event` (`indico.modules.events.notes.models.notes.EventNote` attribute), 190  
`event` (`indico.modules.events.papers.fields.PaperEmailSettingsField` attribute), 333  
`event` (`indico.modules.events.papers.models.competences.PaperCompetence` attribute), 195  
`event` (`indico.modules.events.papers.models.papers.Paper` attribute), 196  
`event` (`indico.modules.events.papers.models.review_questions.PaperReviewQuestion` attribute), 197  
`event` (`indico.modules.events.papers.models.templates.PaperTemplate` attribute), 202  
`event` (`indico.modules.events.registration.models.forms.RegistrationForm` attribute), 219  
`event` (`indico.modules.events.registration.models.registrations.Registration` attribute), 213  
`event` (`indico.modules.events.reminders.models.reminders.EventReminder` attribute), 235  
`event` (`indico.modules.events.sessions.models.blocks.SessionBlock` attribute), 242  
`event` (`indico.modules.events.sessions.models.sessions.Session` attribute), 240

event (indico.modules.events.static.models.static.StaticSite attribute), 262	event_id (indico.modules.events.layout.models.menu.EventPage attribute), 183
event (indico.modules.events.surveys.models.surveys.Survey attribute), 246	event_id (indico.modules.events.layout.models.menu.MenuEntry attribute), 183
event (indico.modules.events.timetable.models.breaks.Break attribute), 253	event_id (indico.modules.events.logs.models.entries.EventLogEntry attribute), 187
event (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254	event_id (indico.modules.events.models.persons.EventPerson attribute), 128
event (indico.modules.events.tracks.models.tracks.Track attribute), 259	event_id (indico.modules.events.models.persons.EventPersonLink attribute), 129
event (indico.modules.events.util.ListGeneratorBase attribute), 139	event_id (indico.modules.events.models.principals.EventPrincipal attribute), 130
event (indico.modules.rb.models.reservations.Reservation attribute), 298	event_id (indico.modules.events.models.references.EventReference attribute), 131
event (indico.modules.rb.models.reservations.ReservationLink attribute), 299	event_id (indico.modules.events.models.settings.EventSetting attribute), 135
event (indico.modules.search.base.SearchTarget attribute), 70	event_id (indico.modules.events.models.settings.EventSettingPrincipal attribute), 136
event (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 313	event_id (indico.modules.events.models.settings.EventSettingsMixin attribute), 136
event (indico.modules.vc.models.vc_rooms.VCRoomLinkType attribute), 313	event_id (indico.modules.events.models.static_list_links.StaticListLink attribute), 137
event_backref_name (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion attribute), 153	event_id (indico.modules.events.notes.models.notes.EventNote attribute), 190
event_backref_name (indico.modules.events.papers.models.review_questions_and_papers.ReviewQuestionsAndPapers attribute), 197	event_id (indico.modules.events.papers.models.competences.PaperCompetence attribute), 195
event_creation_notification_emails (indico.modules.categories.models.categories.Category attribute), 264	event_id (indico.modules.events.papers.models.templates.PaperTemplate attribute), 202
event_creation_restricted (indico.modules.categories.models.categories.Category attribute), 264	event_id (indico.modules.events.registration.models.forms.RegistrationForm attribute), 219
event_id (indico.modules.attachments.models.folders.AttachmentsFolder attribute), 284	event_id (indico.modules.events.registrations.Registration attribute), 213
event_id (indico.modules.designer.models.templates.DesignerTemplate attribute), 317	event_id (indico.modules.events.reminders.models.reminders.EventReminder attribute), 235
event_id (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 145	event_id (indico.modules.events.requests.models.requests.Request attribute), 236
event_id (indico.modules.events.abstracts.models.email_templates.AlertEmailTemplate attribute), 150	event_id (indico.modules.events.sessions.models.sessions.Session attribute), 240
event_id (indico.modules.events.abstracts.models.review_questions_and_papers.ReviewQuestionsAndPapers attribute), 153	event_id (indico.modules.events.static.models.static.StaticSite attribute), 262
event_id (indico.modules.events.agreements.models.agreements.Agreement attribute), 166	event_id (indico.modules.events.surveys.models.surveys.Survey attribute), 246
event_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 170	event_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 254
event_id (indico.modules.events.contributions.models.fields_and_contributors.FieldsAndContributors attribute), 173	event_id (indico.modules.events.tracks.models.tracks.Track attribute), 259
event_id (indico.modules.events.contributions.models.types_and_contributors.TypesAndContributors attribute), 179	event_id (indico.modules.rb.models.reservations.ReservationLink attribute), 299
event_id (indico.modules.events.layout.models.images.ImageFile attribute), 182	event_id (indico.modules.search.result_schemas.AttachmentResultSchema attribute), 72
	event_id (indico.modules.search.result_schemas.ContributionResultSchema attribute), 71

<b>Index</b>	<b>411</b>
--------------	------------

	<i>dico.modules.events.logs.renderers</i> ), 188			<i>dico.modules.events.settings</i> ), 143, 205	
EventMessageMode	(class in <i>dico.modules.categories.models.categories</i> ), 266	in-		EventSettingsMixin	(class in <i>dico.modules.events.models.settings</i> ), 136
EventNote	(class in <i>dico.modules.events.notes.models.notes</i> ), 190	in-		EventSettingsProxy	(class in <i>dico.modules.events.settings</i> ), 143, 205
EventNoteResultSchema	(class in <i>dico.modules.search.result_schemas</i> ), 73	in-		EventSpeakersPlaceholder	(class in <i>dico.modules.designer.placeholders</i> ), 324
EventNoteRevision	(class in <i>dico.modules.events.notes.models.notes</i> ), 192	in-		EventTitlePlaceholder	(class in <i>dico.modules.designer.placeholders</i> ), 323
EventOrgTextPlaceholder	(class in <i>dico.modules.designer.placeholders</i> ), 320	in-		EventTitlePlaceholder	(class in <i>dico.modules.events.abstracts.placeholders</i> ), 159
EventPage	(class in <i>dico.modules.events.layout.models.menu</i> ), 183	in-		EventTitlePlaceholder	(class in <i>dico.modules.events.persons.placeholders</i> ), 211
EventPerson	(class in <i>dico.modules.events.models.persons</i> ), 128	in-		EventTitlePlaceholder	(class in <i>dico.modules.events.registration.placeholders.registrations</i> ), 229
EventPersonLink	(class in <i>dico.modules.events.models.persons</i> ), 129	in-		EventType	(class in <i>dico.modules.events.models.events</i> ), 127
EventPersonLinkListField	(class in <i>dico.modules.events.fields</i> ), 327	in-		EventURLPlaceholder	(class in <i>dico.modules.events.abstracts.placeholders</i> ), 159
EventPersonListField	(class in <i>dico.modules.events.fields</i> ), 327	in-		EventVenuePlaceholder	(class in <i>dico.modules.designer.placeholders</i> ), 324
EventPrincipal	(class in <i>dico.modules.events.models.principals</i> ), 130	in-		EXPERIMENTAL_EDITING_SERVICE	(built-in variable), 54
EventReference	(class in <i>dico.modules.events.models.references</i> ), 131	in-		expired	( <i>indico.modules.events.static.models.static.StaticSiteState</i> attribute), 263
EventReminder	(class in <i>dico.modules.events.reminders.models.reminders</i> ), 234	in-		extend_defaults()	(in <i>dico.modules.users.ext.ExtraUserPreferences</i> method), 280
EventResultSchema	(class in <i>dico.modules.search.result_schemas</i> ), 71	in-		extend_end_dt()	(in <i>dico.modules.events.timetable.models.entries.TimetableEntry</i> method), 254
EventRoomPlaceholder	(class in <i>dico.modules.designer.placeholders</i> ), 324	in-		extend_form()	(in <i>dico.modules.users.ext.ExtraUserPreferences</i> method), 280
events_backref_name	(in <i>dico.modules.attachments.models.folders.AttachmentFolder</i> attribute), 284	in-		extend_parent()	(in <i>dico.modules.events.timetable.models.entries.TimetableEntry</i> method), 254
events_backref_name	(in <i>dico.modules.events.notes.models.notes.EventNote</i> attribute), 190	in-		extend_start_dt()	(in <i>dico.modules.events.timetable.models.entries.TimetableEntry</i> method), 255
events_backref_name	(in <i>dico.modules.rb.models.reservations.ReservationLink</i> attribute), 299	in-		extension	( <i>indico.modules.attachments.models.attachments.Attachment</i> attribute), 283
EventSeries	(class in <i>dico.modules.events.models.series</i> ), 135	in-		extension	( <i>indico.modules.designer.models.images.DesignerImageFile</i> attribute), 317
EventSetting	(class in <i>dico.modules.events.models.settings</i> ), 135	in-		extension	( <i>indico.modules.events.abstracts.models.files.AbstractFile</i> attribute), 152
EventSettingPrincipal	(class in <i>dico.modules.events.models.settings</i> ), 135	in-		extension	( <i>indico.modules.events.layout.models.images.ImageFile</i> attribute), 182
EventSettingProperty	(class in <i>dico.modules.events.papers.models.files.PaperFile</i> attribute), 195	in-		extension	( <i>indico.modules.events.papers.models.files.PaperFile</i> attribute), 195





dico.modules.events.registration.models.registration	indico.modules.events.abstrac
attribute), 215	attribute), 152
field_id (indico.modules.events.registration.models.form_fields.RegistrationFormFieldData.layout.models.images.ImageField	attribute), 182
attribute), 218	
field_impl (indico.modules.events.registration.models.form_fields.RegistrationFormFieldData.papers.models.files.PaperFile	attribute), 195
attribute), 217	
field_pd (indico.modules.events.registration.models.items.RegistrationFormFieldData.events.papers.models.templates.PaperTemplate	attribute), 202
attribute), 224	
field_type (indico.modules.events.abstrac	indico.modules.events.registration.models.registrations.Registr
attribute), 153	attribute), 216
field_type (indico.modules.events.contributions.models.fields.ContributionFieldData.modules.events.static.models.static.StaticSite	attribute), 262
attribute), 173	
field_type (indico.modules.events.papers.models.review_questions.PublicReviewQuestionArch.result_schemas.AttachmentResultSchem	attribute), 72
attribute), 197	
field_type (indico.modules.events.surveys.models.items.SurveyItemData.modules.events.papers.models.papers.Paper	attribute), 196
attribute), 248	
field_type (indico.modules.events.surveys.models.items.SurveyQuestionData.modules.search.result_schemas.BucketSchema	attribute), 68
attribute), 249	
field_type (indico.modules.events.surveys.models.items.SurveySectionData.modules.events.papers.models.papers.Paper	attribute), 290
attribute), 249	
field_type (indico.modules.events.surveys.models.items.SurveyTextData.modules.events.papers.models.papers.Paper	attribute), 250
attribute), 250	
field_values (indico.modules.events.abstrac	indico.modules.rb.models.rooms.Room
attribute), 145	static
field_values (indico.modules.events.contributions.models.fields.ContributionFieldData.modules.events.contributions.models.fields.ContributionField	attribute), 173
FieldPlaceholder (class in in-	indico.modules.events.registration.placeholders.registrations), 230
fields (indico.modules.events.registration.models.items.RegistrationFormSectionData.modules.events.abstrac	indico.modules.events.abstrac
attribute), 225	indico.modules.events.abstrac
fields (indico.modules.users.ext.ExtraUserPreferences	indico.modules.rb.models.rooms.Room
attribute), 280	static
FieldStats (class in in-	indico.modules.rb.models.rooms.Room
dico.modules.events.registration.stats), 233	static
file (indico.modules.attachments.models.attachments.AttachmentData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 281	indico.modules.events.papers.models.papers.Paper
file (indico.modules.attachments.models.attachments.AttachmentTypeData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 283	indico.modules.events.papers.models.papers.Paper
file (indico.modules.events.registration.models.registration.RegistrationData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 216	indico.modules.events.papers.models.papers.Paper
file_id (indico.modules.attachments.models.attachments.AttachmentData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 281	indico.modules.events.papers.models.papers.Paper
file_required (in-	indico.modules.events.papers.models.papers.Paper
dico.modules.events.registration.models.registrations.RegistrationData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 216	indico.modules.events.papers.models.papers.Paper
file_required (in-	indico.modules.events.papers.models.papers.Paper
dico.modules.events.static.models.static.StaticSiteData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 262	indico.modules.events.papers.models.papers.Paper
FormField (class in indico.web.forms.fields), 342	indico.modules.events.papers.models.papers.Paper
filename (indico.modules.attachments.models.attachments.AttachmentData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 283	indico.modules.events.papers.models.papers.Paper
filename (indico.modules.designer.models.images.DesignerImageFileData.modules.events.papers.models.papers.Paper	indico.modules.events.papers.models.papers.Paper
attribute), 317	indico.modules.events.papers.models.papers.Paper

method), 298

find\_overlapping\_with() (in- dico.modules.rb.models.reservation\_occurrences.ReservationOccurrence class method), 300

find\_overlapping\_with() (in- dico.modules.rb.models.reservations.Reservation class static method), 298

find\_with\_attribute() (in- dico.modules.rb.models.rooms.Room class method), 291

finished(indico.modules.events.surveys.models.surveys.SurveyState attribute), 247

first\_last(indico.modules.users.models.users.NameFormat attribute), 271

first\_last\_upper(indico.modules.users.models.users.NameFormat attribute), 271

first\_name(indico.modules.events.models.persons.EventPerson attribute), 128

first\_name(indico.modules.events.models.persons.PersonLinkBase attribute), 130

first\_name(indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 222

first\_name(indico.modules.events.registration.models.items.PersonalDataTypes attribute), 223

first\_name(indico.modules.events.registration.models.registrations.Registration attribute), 213

first\_name(indico.modules.users.models.users.User attribute), 273

FirstNamePlaceholder(class in in- dico.modules.events.persons.placeholders), 211

FirstNamePlaceholder(class in in- dico.modules.events.registration.placeholders.invitations), 231

FirstNamePlaceholder(class in in- dico.modules.events.registration.placeholders.registrations), 230

fit\_session\_block\_entry()(in module in- dico.modules.events.timetable.operations), 256

fits\_period()(in- dico.modules.rb.models.room\_bookable\_hours.BookableHours class method), 293

flash\_info\_message()(in- dico.modules.events.util.ListGeneratorBase class method), 139

floor(indico.modules.rb.models.rooms.Room attribute), 291

folder(indico.modules.attachments.models.attachments.Attachment attribute), 281

folder\_created(in module in- dico.core.signals.attachments), 82

folder\_deleted(in module in- dico.core.signals.attachments), 82

folder\_id(indico.modules.attachments.models.attachments.Attachment attribute), 282

folder\_id(indico.modules.attachments.models.principals.AttachmentForPrincipal attribute), 286

folder\_id(indico.modules.search.result\_schemas.AttachmentResultSchema attribute), 72

folder\_updated(in module in- dico.core.signals.attachments), 82

for\_user()(indico.modules.events.models.persons.EventPerson class method), 128

form(indico.modules.events.requests.base.RequestDefinitionBase attribute), 238

form\_defaults(in- dico.modules.events.requests.base.RequestDefinitionBase attribute), 238

form\_items(indico.modules.events.registration.models.forms.RegistrationForm attribute), 219

form\_validated(in module in- dico.core.signals.core), 83

format\_display\_full\_name()(in module in- dico.modules.events.registration.models.users), 276

format\_feature\_names()(in module in- dico.modules.events.features.util), 181

frame(indico.modules.events.abstracts.settings.BOALinkFormat attribute), 166

friendly\_data(in- dico.modules.events.contributions.models.fields.ContributionField attribute), 174

friendly\_data(in- dico.modules.events.registration.models.registrations.Registration attribute), 216

friendly\_id(indico.modules.events.abstracts.models.abstracts.Abstract attribute), 145

friendly\_id(indico.modules.events.contributions.models.contributions.Contribution attribute), 170

friendly\_id(indico.modules.events.contributions.models.subcontributions.Subcontribution attribute), 177

friendly\_id(indico.modules.events.registration.models.registrations.Registration attribute), 213

friendly\_id(indico.modules.events.sessions.models.sessions.Session attribute), 240

friendly\_id(indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 251

friendly\_name(in- dico.modules.vc.plugins.VCPluginMixin attribute), 315

full\_access(indico.modules.categories.models.principals.CategoryPrincipal attribute), 267

full\_access(indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 176

full\_access(indico.modules.events.models.principals.EventPrincipal attribute), 131

full\_access(indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 176

[attribute](#)), 243  
[full\\_access \(indico.modules.events.tracks.models.principals.TrackPrincipal\)](#) (in module [indico.modules.events.tracks.models.principals](#)), 261  
[full\\_name \(indico.modules.events.registration.models.registration.Registration\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[full\\_name \(indico.modules.rb.models.rooms.Room\)](#) (in module [indico.modules.rb.models.rooms](#)), 291  
[full\\_name \(indico.modules.users.models.users.PersonMixin\)](#) (in module [indico.modules.users.models.users](#)), 271  
[full\\_title \(indico.modules.events.models.reviews.ProposalGroupProxy\)](#) (in module [indico.modules.events.models.reviews](#)), 133  
[full\\_title \(indico.modules.events.sessions.models.blocks.SessionBlock\)](#) (in module [indico.modules.events.sessions.models.blocks](#)), 242  
[full\\_title \(indico.modules.events.tracks.models.tracks.Track\)](#) (in module [indico.modules.events.tracks.models.tracks](#)), 259  
[full\\_title\\_attr](#) (in module [indico.modules.events.models.reviews.ProposalGroupProxy](#)), 133  
[full\\_title\\_with\\_group](#) (in module [indico.modules.events.tracks.models.tracks.Track](#)), 259  
**G**  
[generate\\_content \(\)](#) (in module [indico.modules.attachments.preview.MarkdownPreviewer](#)), 288  
[generate\\_content \(\)](#) (in module [indico.modules.attachments.preview.Previewer](#)), 289  
[generate\\_content \(\)](#) (in module [indico.modules.attachments.preview.TextPreviewer](#)), 289  
[generate\\_name \(\)](#) (in module [indico.modules.rb.models.rooms.Room](#)), 291  
[generate\\_pdf\\_from\\_sessions \(\)](#) (in module [indico.modules.events.sessions.util](#)), 245  
[generate\\_spreadsheet\\_from\\_abstracts \(\)](#) (in module [indico.modules.events.abstracts.util](#)), 157  
[generate\\_spreadsheet\\_from\\_contributions \(\)](#) (in module [indico.modules.events.contributions.util](#)), 180  
[generate\\_spreadsheet\\_from\\_occurrences \(\)](#) (in module [indico.modules.rb.util](#)), 301  
[generate\\_spreadsheet\\_from\\_registrations \(\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[generate\\_spreadsheet\\_from\\_sessions \(\)](#) (in module [indico.modules.events.sessions.util](#)), 245  
[generate\\_spreadsheet\\_from\\_survey \(\)](#) (in module [indico.modules.events.surveys.util](#)), 252  
[generate\\_static\\_url \(\)](#) (in module [indico.modules.events.util.ListGeneratorBase](#)), 139  
[get \(\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[get \(\)](#) (in module [indico.modules.events.registration.util](#)), 227  
[get \(\)](#) (in module [indico.modules.categories.settings.CategorySettingsProxy](#)), 270  
[get \(\)](#) (in module [indico.modules.events.settings.EventACLProxy](#)), 142, 205  
[get \(\)](#) (in module [indico.modules.events.settings.EventSettingsProxy](#)), 143, 206  
[get \(\)](#) (in module [indico.modules.users.models.settings.UserSettingsProxy](#)), 278  
[get\\_access\\_list \(\)](#) (in module [indico.modules.events.contributions.models.subcontributions.SubContribution](#)), 177  
[get\\_active\\_payment\\_plugins \(\)](#) (in module [indico.modules.events.payment.util](#)), 208  
[get\\_admin\\_emails \(\)](#) (in module [indico.modules.users.util](#)), 279  
[get\\_agreement\\_definitions \(\)](#) (in module [indico.modules.events.agreements.util](#)), 167  
[get\\_all \(\)](#) (in module [indico.modules.categories.settings.CategorySettingsProxy](#)), 270  
[get\\_all \(\)](#) (in module [indico.modules.events.settings.EventSettingsProxy](#)), 143, 206  
[get\\_all \(\)](#) (in module [indico.modules.users.models.settings.UserSettingsProxy](#)), 278  
[get\\_all\\_for\\_event \(\)](#) (in module [indico.modules.events.registration.models.registrations.Registration](#)), 213  
[get\\_all\\_templates \(\)](#) (in module [indico.modules.designer.util](#)), 318  
[get\\_all\\_user\\_roles \(\)](#) (in module [indico.modules.events.util](#)), 139  
[get\\_allowed\\_scope \(\)](#) (in module [indico.core.oauth.models.applications.OAuthApplication](#)), 306  
[get\\_allowed\\_sender\\_emails \(\)](#) (in module [indico.modules.events.models.events.Event](#)), 123  
[get\\_attached\\_folders \(\)](#) (in module [indico.modules.attachments.util](#)), 287  
[get\\_attached\\_items \(\)](#) (in module [indico.modules.attachments.util](#)), 288  
[get\\_attachment\\_count \(\)](#) (in module [indico.modules.categories.util](#)), 268  
[get\\_attribute\\_by\\_name \(\)](#) (in module [indico.modules.rb.models.rooms.Room](#)), 291



[get\\_attribute\\_value\(\)](#) (in *indico.modules.rb.models.rooms.Room* method), 291  
[get\\_auth\\_time\(\)](#) (in *indico.core.oauth.models.tokens.OAuth2AuthorizationCode* method), 308  
[get\\_avatar\\_url\\_from\\_name\(\)](#) (in module *indico.modules.users.util*), 279  
[get\\_badge\\_format\(\)](#) (in module *indico.modules.designer.util*), 318  
[get\\_blocked\\_rooms\(\)](#) (in *indico.modules.rb.models.rooms.Room* method), 291  
[get\\_blueprints](#) (in module *indico.core.signals.plugin*), 88  
[get\\_blueprints\(\)](#) (in *indico.core.plugins.IndicoPlugin* method), 78  
[get\\_boa\\_export\\_formats\(\)](#) (in module *indico.modules.events.contributions.util*), 180  
[get\\_booking\\_params\\_for\\_event\(\)](#) (in module *indico.modules.rb.util*), 301  
[get\\_category\\_stats\(\)](#) (in module *indico.modules.categories.util*), 268  
[get\\_category\\_timetable\(\)](#) (in module *indico.modules.events.timetable.util*), 257  
[get\\_client\\_id\(\)](#) (in *indico.core.oauth.models.applications.OAuthApplication* method), 306  
[get\\_cloners](#) (in module *indico.core.signals.event\_management*), 87  
[get\\_color\\_for\\_user\\_id\(\)](#) (in module *indico.modules.users.util*), 279  
[get\\_conditions](#) (in module *indico.core.signals.core*), 83  
[get\\_conference\\_themes](#) (in module *indico.core.signals.plugin*), 88  
[get\\_conflicting\\_occurrences\(\)](#) (in *indico.modules.rb.models.reservations.Reservation* method), 298  
[get\\_contribs\\_by\\_year\(\)](#) (in module *indico.modules.categories.util*), 268  
[get\\_contribution\(\)](#) (in *indico.modules.events.models.events.Event* method), 124  
[get\\_contribution\\_field\(\)](#) (in *indico.modules.events.models.events.Event* method), 124  
[get\\_contributions\\_for\\_person\(\)](#) (in module *indico.modules.events.contributions.util*), 180  
[get\\_contributions\\_with\\_paper\\_submitted\\_by\\_event\(\)](#) (in module *indico.modules.events.papers.util*), 204  
[get\\_contributions\\_with\\_user\\_as\\_submitter\(\)](#) (in module *indico.modules.events.contributions.util*), 180  
[get\\_css\\_file\\_data\(\)](#) (in module *indico.modules.events.layout.util*), 186  
[get\\_css\\_url\(\)](#) (in module *indico.modules.events.layout.util*), 186  
[get\\_data\(\)](#) (*indico.modules.events.logs.renderers.EventLogRendererBase* class method), 188  
[get\\_data\(\)](#) (*indico.modules.events.logs.renderers.SimpleRenderer* class method), 189  
[get\\_default\\_badge\\_on\\_category\(\)](#) (in module *indico.modules.designer.util*), 318  
[get\\_default\\_folder\\_names\(\)](#) (in module *indico.modules.attachments.util*), 288  
[get\\_default\\_redirect\\_uri\(\)](#) (in *indico.core.oauth.models.applications.OAuthApplication* method), 306  
[get\\_default\\_ticket\\_on\\_category\(\)](#) (in module *indico.modules.designer.util*), 318  
[get\\_definitions](#) (in module *indico.core.signals.agreements*), 82  
[get\\_delete\\_comment\\_url\(\)](#) (in *indico.modules.events.models.reviews.ProposalMixin* method), 134  
[get\\_disallowed\\_features\(\)](#) (in module *indico.modules.events.features.util*), 181  
[get\\_download\\_url\(\)](#) (in *indico.modules.attachments.models.attachments.Attachment* method), 282  
[get\\_editable\(\)](#) (in *indico.modules.events.contributions.models.contributions.Contribution* method), 170  
[get\\_enabled\\_features\(\)](#) (in module *indico.modules.events.features.util*), 181  
[get\\_event\(\)](#) (in module *indico.modules.attachments.util*), 288  
[get\\_event\\_from\\_url\(\)](#) (in module *indico.modules.events.util*), 139  
[get\\_event\\_management\\_url\(\)](#) (in *indico.modules.events.payment.plugins.PaymentPluginMixin* method), 209  
[get\\_event\\_regforms\(\)](#) (in module *indico.modules.events.registration.util*), 227  
[get\\_event\\_regforms\\_registrations\(\)](#) (in module *indico.modules.events.registration.util*), 228  
[get\\_event\\_request\\_definitions](#) (in module *indico.core.signals.plugin*), 88  
[get\\_event\\_section\\_data\(\)](#) (in module *indico.modules.events.registration.util*), 228  
[get\\_event\\_themes\\_files](#) (in module *indico.core.signals.plugin*), 88  
[get\\_events\\_by\\_year\(\)](#) (in module *indico.modules.categories.util*), 268

`get_events_created_by()` (in module `indico.modules.events.util`), 140  
`get_events_managed_by()` (in module `indico.modules.events.util`), 140  
`get_events_registered()` (in module `indico.modules.events.registration.util`), 228  
`get_events_with_abstract_persons()` (in module `indico.modules.events.abstracts.util`), 158  
`get_events_with_abstract_reviewer_convener()` (in module `indico.modules.events.abstracts.util`), 158  
`get_events_with_linked_contributions()` (in module `indico.modules.events.contributions.util`), 180  
`get_events_with_linked_event_persons()` (in module `indico.modules.events.util`), 140  
`get_events_with_linked_sessions()` (in module `indico.modules.events.sessions.util`), 245  
`get_events_with_paper_roles()` (in module `indico.modules.events.papers.util`), 204  
`get_events_with_submitted_surveys()` (in module `indico.modules.events.surveys.util`), 252  
`get_expires_in()` (in `indico.core.oauth.models.tokens.TokenModelBase` method), 309  
`get_extra_delete_msg()` (in `indico.modules.vc.plugins.VCPluginMixin` method), 315  
`get_feature_definition()` (in module `indico.modules.events.features.util`), 181  
`get_feature_definitions` (in module `indico.core.signals.event`), 85  
`get_feature_definitions()` (in module `indico.modules.events.features.util`), 181  
`get_field_value()` (in `indico.modules.events.contributions.models.contributions.CustomFieldsMixin` method), 172  
`get_field_values()` (in module `indico.modules.events.util`), 140  
`get_fields` (in module `indico.core.signals.core`), 83  
`get_file_previewer()` (in module `indico.modules.attachments.preview`), 289  
`get_file_previewers` (in module `indico.core.signals.attachments`), 82  
`get_file_previewers()` (in module `indico.modules.attachments.preview`), 289  
`get_for_event()` (in `indico.modules.events.layout.models.menu.MenuEntry` static method), 183  
`get_for_linked_object()` (in `indico.modules.attachments.models.folders.AttachmentFolder` class method), 284  
`get_for_linked_object()` (in `indico.modules.events.notes.models.notes.EventNote` class method), 190  
`get_friendly_data()` (in `indico.modules.events.registration.models.form_fields.RegistrationFormFieldsMixin` method), 217  
`get_friendly_data()` (in `indico.modules.events.registration.models.registrations.RegistrationFormFieldsMixin` method), 216  
`get_full_name()` (in `indico.modules.events.registration.models.registrations.RegistrationFormFieldsMixin` method), 213  
`get_full_name()` (in `indico.modules.users.models.users.PersonMixin` method), 271  
`get_full_name()` (in `indico.modules.users.models.users.User` method), 273  
`get_gravatar_for_user()` (in module `indico.modules.users.util`), 279  
`get_hidden_events()` (in `indico.modules.categories.models.categories.Category` method), 264  
`get_highest` (`indico.modules.events.contributions.models.persons.Authors` attribute), 174  
`get_icon_data_cte()` (in `indico.modules.categories.models.categories.Category` class method), 264  
`get_identity()` (in `indico.modules.users.models.users.User` method), 273  
`get_image_data()` (in module `indico.modules.categories.util`), 269  
`get_image_placeholder_types()` (in module `indico.modules.designer.util`), 318  
`get_inherited_templates()` (in module `indico.modules.designer.util`), 318  
`get_inherited_templates()` (in `indico.modules.events.payment.plugins.PaymentPluginMixin` method), 209  
`get_label_markup()` (in `indico.modules.events.models.events.Event` method), 124  
`get_last_revision()` (in `indico.modules.events.models.reviews.ProposalMixin` method), 134  
`get_last_revision()` (in `indico.modules.events.papers.models.papers.Paper` method), 196  
`get_linked_events()` (in module `indico.modules.users.util`), 279  
`get_linked_for_event()` (in `indico.modules.vc.models.vc_rooms.VCRoomEventAssociation` class method), 284

- class method*), 313
- `get_linked_object()` (in module *indico.modules.rb.util*), 301
- `get_linked_to_description()` (in module *indico.modules.vc.util*), 314
- `get_list_url()` (in *indico.modules.events.util.ListGeneratorBase* *method*), 139
- `get_log_renderers` (in module *indico.core.signals.event*), 85
- `get_log_renderers()` (in module *indico.modules.events.logs.util*), 188
- `get_logo_data()` (in module *indico.modules.events.layout.util*), 186
- `get_managed_vc_plugins()` (in module *indico.modules.vc.util*), 314
- `get_management_permissions` (in module *indico.core.signals.acl*), 81
- `get_manager_list()` (in *indico.modules.events.contributions.models.subcontributions.SubContribution* *method*), 177
- `get_manager_notification_emails()` (in *indico.modules.events.requests.base.RequestDefinitionBase* *class method*), 238
- `get_members()` (in *indico.modules.groups.core.GroupProxy* *method*), 311
- `get_menu_entries_from_signal()` (in module *indico.modules.events.layout.util*), 186
- `get_menu_entry_by_name()` (in module *indico.modules.events.layout.util*), 186
- `get_merged_from_users_recursive()` (*indico.modules.users.models.users.User* *method*), 273
- `get_message()` (in *indico.modules.rb.models.reservations.RepeatMapping* *class method*), 296
- `get_method_name()` (in *indico.modules.events.payment.plugins.PaymentPluginMixin* *method*), 209
- `get_min_year()` (in module *indico.modules.categories.util*), 269
- `get_nested_entries()` (in module *indico.modules.events.timetable.util*), 258
- `get_nested_placeholder_options()` (in module *indico.modules.designer.util*), 318
- `get_next_position()` (in module *indico.modules.events.tracks.models.tracks*), 260
- `get_non_inheriting_objects()` (in *indico.modules.events.contributions.models.contributions.Contribution* *method*), 170
- `get_non_inheriting_objects()` (in *indico.modules.events.models.events.Event* *method*), 124
- `get_non_inheriting_objects()` (in *indico.modules.events.sessions.models.sessions.Session* *method*), 240
- `get_nonce()` (*indico.core.oauth.models.tokens.OAuth2AuthorizationCode* *method*), 308
- `get_not_deletable_templates()` (in module *indico.modules.designer.util*), 318
- `get_notification_bcc_list()` (in *indico.modules.vc.plugins.VCPluginMixin* *method*), 315
- `get_notification_cc_list()` (in *indico.modules.vc.plugins.VCPluginMixin* *method*), 315
- `get_notification_reply_email()` (in *indico.modules.events.requests.base.RequestDefinitionBase* *class method*), 238
- `get_notification_template()` (in *indico.modules.events.requests.base.RequestDefinitionBase* *class method*), 238
- `get_object_from_args()` (in module *indico.modules.events.util*), 140
- `get_or_create()` (in *indico.modules.attachments.models.folders.AttachmentFolder* *class method*), 284
- `get_or_create_default()` (in *indico.modules.attachments.models.folders.AttachmentFolder* *class method*), 284
- `get_overlap()` (in *indico.modules.rb.models.reservation\_occurrences.ReservationOccurrence* *method*), 300
- `get_overridden_value()` (in *indico.web.forms.fields.OverrideMultipleItemsField* *method*), 343
- `get_participant_list_columns()` (in *indico.modules.events.registration.settings.RegistrationSettingsProxy* *method*), 232
- `get_participant_list_form_ids()` (in *indico.modules.events.registration.settings.RegistrationSettingsProxy* *method*), 232
- `get_payment_plugins()` (in module *indico.modules.events.payment.util*), 208
- `get_pdf()` (*indico.modules.designer.pdf.DesignerPDFBase* *method*), 318
- `get_permissions_for_user()` (in *indico.modules.rb.models.rooms.Room* *class method*), 291
- `get_personal_data()` (in *indico.modules.events.registration.models.registrations.Registration* *method*), 213
- `get_personal_data_field_id()` (in

*dico.modules.events.registration.models.forms.RegistrationForm* method), 134

*method*), 219

*get\_placeholder\_options()* (in module *indico.modules.designer.util*), 318

*get\_placeholders* (in module *indico.core.signals.core*), 83

*get\_placeholders()* (in *dico.modules.search.base.IndicoSearchProvider* method), 69, 70

*get\_plugin\_conference\_themes()* (in module *indico.modules.events.layout.util*), 186

*get\_plugin\_template\_module()* (in module *indico.core.plugins*), 80

*get\_prebooking\_collisions()* (in module *indico.modules.rb.util*), 301

*get\_protection\_cte()* (in *dico.modules.categories.models.categories.Category* class method), 264

*get\_protection\_parent\_cte()* (in *dico.modules.categories.models.categories.Category* method), 264

*get\_published\_registrations()* (in module *indico.modules.events.registration.util*), 228

*get\_questions\_for\_review\_type()* (in *dico.modules.events.papers.models.call\_for\_papers.CallForPapers* method), 193

*get\_random\_color()* (in module *indico.modules.events.util*), 140

*get\_recent\_news()* (in module *indico.modules.news.util*), 327

*get\_redirect\_uri()* (in *dico.core.oauth.models.tokens.OAuth2AuthorizationCode* method), 308

*get\_registered\_event\_persons()* (in module *indico.modules.events.registration.util*), 228

*get\_registration()* (in *dico.modules.events.registration.models.forms.RegistrationForm* method), 220

*get\_registrations\_with\_tickets()* (in module *indico.modules.events.registration.util*), 228

*get\_related\_categories()* (in module *indico.modules.users.util*), 279

*get\_relative\_event\_ids()* (in *dico.modules.events.models.events.Event* method), 124

*get\_request\_definitions()* (in module *indico.modules.events.requests.util*), 237

*get\_resized\_room\_photo()* (in module *indico.modules.rb.util*), 301

*get\_reviewed\_for\_groups()* (in *dico.modules.events.abstracts.models.abstracts.Abstract* method), 145

*get\_reviewed\_for\_groups()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 219

*get\_reviewed\_for\_groups()* (in *dico.modules.events.papers.models.revisions.PaperRevision* method), 201

*get\_reviewer\_render\_data()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_reviewing\_state()* (in *dico.modules.events.papers.models.call\_for\_papers.CallForPapers* method), 193

*get\_reviews()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_reviews()* (in *dico.modules.events.papers.models.revisions.PaperRevision* method), 201

*get\_revisions()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_revisions()* (in *dico.modules.events.papers.models.papers.Paper* method), 196

*get\_root()* (*indico.modules.categories.models.categories.Category* class method), 264

*get\_root()* (*indico.modules.events.papers.models.papers.Paper* method), 196

*get\_save\_comment\_url()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_save\_judgment\_url()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_save\_review\_url()* (in *dico.modules.events.models.reviews.ProposalRevisionMixin* method), 134

*get\_scheduled\_notes()* (in module *indico.modules.events.notes.util*), 192

*get\_scope()* (*indico.core.oauth.models.tokens.OAuth2AuthorizationCode* method), 308

*get\_scope()* (*indico.core.oauth.models.tokens.OAuthToken* method), 309

*get\_search\_provider()* (in module *indico.modules.search.base*), 70

*get\_search\_providers* (in module *indico.core.signals.core*), 84

*get\_session()* (in *dico.modules.events.models.events.Event* method), 124

*get\_session\_block()* (in *dico.modules.events.models.events.Event* method), 124

*get\_session\_block\_entries()* (in module *indico.modules.events.timetable.util*), 258



`get_session_timetable_pdf()` (in module `indico.modules.events.sessions.util`), 245  
`get_sessions_for_user()` (in module `indico.modules.events.sessions.util`), 245  
`get_short_name()` (in module `indico.modules.rb.models.reservations.RepeatMapping` class method), 296  
`get_sibling_entry()` (in module `indico.modules.events.timetable.operations`), 256  
`get_sort_options()` (in module `indico.modules.search.base.IndicoSearchProvider` method), 70  
`get_sorted_tracks()` (in module `indico.modules.events.models.events.Event` method), 124  
`get_spotlight_file()` (in module `indico.modules.events.papers.models.revisions.PaperRevision` method), 201  
`get_storage_backends` (in module `indico.core.signals.core`), 84  
`get_suggested_categories()` (in module `indico.modules.users.util`), 279  
`get_summary()` (in module `indico.modules.events.surveys.models.items.SurveyQuestion` method), 249  
`get_system_user()` (in module `indico.modules.users.models.users.User` static method), 273  
`get_table()` (`indico.modules.events.registration.stats.FieldStats` method), 234  
`get_template_customization_paths` (in module `indico.core.signals.plugin`), 88  
`get_theme()` (in module `indico.modules.events.util`), 140  
`get_themes_for()` (in module `indico.modules.events.settings.ThemeSettingsProxy` method), 144, 206  
`get_ticket_attachments()` (in module `indico.modules.events.registration.util`), 228  
`get_time_changes_notifications()` (in module `indico.modules.events.timetable.util`), 258  
`get_timeline()` (in module `indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`get_timeline()` (in module `indico.modules.events.models.reviews.ProposalRevisionMixin` method), 134  
`get_timeline()` (in module `indico.modules.events.papers.models.revisions.PaperRevision` method), 201  
`get_timetable_offline_pdf_generator()` (in module `indico.modules.events.timetable.util`), 258  
`get_title()` (`indico.modules.events.registration.models.items.Personal` method), 223  
`get_title_uuid()` (in module `indico.modules.events.registration.util`), 228  
`get_top_level_entries()` (in module `indico.modules.events.timetable.util`), 258  
`get_track_question_scores()` (in module `indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`get_track_reviewer_abstract_counts()` (in module `indico.modules.events.abstracts.util`), 158  
`get_track_reviewing_state()` (in module `indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`get_track_score()` (in module `indico.modules.events.abstracts.models.abstracts.Abstract` method), 145  
`get_tree_cte()` (in module `indico.modules.categories.models.categories.Category` class method), 264  
`get_upcoming_events()` (in module `indico.modules.categories.util`), 269  
`get_user_abstracts()` (in module `indico.modules.events.abstracts.util`), 158  
`get_user_by_email()` (in module `indico.modules.users.util`), 279  
`get_user_contributions_to_review()` (in module `indico.modules.events.papers.util`), 204  
`get_user_reviewed_contributions()` (in module `indico.modules.events.papers.util`), 204  
`get_user_submittable_contributions()` (in module `indico.modules.events.papers.util`), 204  
`get_user_tracks()` (in module `indico.modules.events.abstracts.util`), 158  
`get_vars_js()` (`indico.core.plugins.IndicoPlugin` method), 78  
`get_vc_plugins()` (in module `indico.modules.vc.util`), 314  
`get_vc_room_attach_form_defaults()` (`indico.modules.vc.plugins.VCPluginMixin` method), 315  
`get_vc_room_form_defaults()` (`indico.modules.vc.plugins.VCPluginMixin` method), 315  
`get_verbose_title()` (in module `indico.modules.events.models.events.Event` method), 124  
`get_visibility_options()` (in module `indico.modules.categories.util`), 269  
`get_visible_categories_cte()` (in module `indico.modules.categories.models.categories.Category` static method), 265  
`get_visible_reviewed_for_tracks()` (in

module `indico.modules.events.abstracts.util`), **H**  
 158  
`get_with_data()` (in- happens\_between() (in-  
`dico.modules.rb.models.reservations.Reservation` `dico.modules.events.models.events.Event`  
 static method), 298 method), 124  
`get_with_data()` (in- has\_attribute() (in-  
`dico.modules.rb.models.rooms.Room` static `dico.modules.rb.models.rooms.Room` method),  
 method), 291 291  
`getBody()` (`indico.modules.events.sessions.util.SessionListToPDF` has\_conflict() (in-  
 method), 244 `dico.modules.events.registration.models.registrations.Registration`  
 method), 213  
`gravatar(indico.modules.users.models.users.ProfilePictureSource` has\_contributions\_with\_user\_as\_submitter()  
 attribute), 272 (in module in-  
`group(indico.modules.designer.placeholders.CategoryTitlePlaceholder` `dico.modules.events.contributions.util`), 180  
 attribute), 324 has\_contributions\_with\_user\_paper\_submission\_rights()  
`group(indico.modules.designer.placeholders.EventDatesPlaceholder` (in module `indico.modules.events.papers.util`),  
 attribute), 319 204  
`group(indico.modules.designer.placeholders.EventDescriptionPlaceholder` has\_custom\_boa (in-  
 attribute), 319 `dico.modules.events.models.events.Event`  
`group(indico.modules.designer.placeholders.EventLogoPlaceholder` attribute), 124  
 attribute), 325  
`group(indico.modules.designer.placeholders.EventOrgTextPlaceholder` has\_effective\_icon (in-  
 attribute), 320 `dico.modules.categories.models.categories.Category`  
 attribute), 265  
`group(indico.modules.designer.placeholders.EventRoomPlaceholder` has\_ended(`indico.modules.events.abstracts.models.call_for_abstracts.C`  
 attribute), 324 attribute), 148  
`group(indico.modules.designer.placeholders.EventSpeakersPlaceholder` has\_ended(`indico.modules.events.models.events.Event`  
 attribute), 324 attribute), 124  
`group(indico.modules.designer.placeholders.EventTitlePlaceholder` has\_ended(`indico.modules.events.papers.models.call_for_papers.CallFo`  
 attribute), 323 attribute), 193  
`group(indico.modules.designer.placeholders.EventVenuePlaceholder` has\_ended(`indico.modules.events.registration.models.forms.Registration`  
 attribute), 324 attribute), 220  
`group(indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder` has\_ended(`indico.modules.events.surveys.models.surveys.Survey`  
 attribute), 321 attribute), 246  
`group(indico.modules.events.models.reviews.ProposalReviewMixin` has\_equipment() (in-  
 attribute), 134 `dico.modules.rb.models.rooms.Room` method),  
`group(indico.modules.groups.core.GroupProxy` at- 291  
 tribute), 311  
`group_attr(indico.modules.events.abstracts.models.reviews.AbstractReview` has\_feature() (in-  
 attribute), 155 `dico.modules.events.models.events.Event`  
 method), 124  
`group_attr(indico.modules.events.models.reviews.ProposalReviewMixin` has\_files(`indico.modules.events.registration.models.registrations.Regis`  
 attribute), 134 attribute), 213  
`group_attr(indico.modules.events.papers.models.reviews.PaperReview` has\_icon(`indico.modules.categories.models.categories.Category`  
 attribute), 199 attribute), 265  
`group_by_occurrence_date()` (in module in- has\_logo(`indico.modules.categories.models.categories.Category`  
`dico.modules.rb.util`), 301 attribute), 265  
`group_id(indico.modules.networks.models.networks.IPNetwork` has\_logo(`indico.modules.events.models.events.Event`  
 attribute), 325 attribute), 124  
`group_proxy_cls` (in- has\_member() (`indico.modules.groups.core.GroupProxy`  
`dico.modules.events.models.reviews.ProposalReviewMixin` method), 311  
 attribute), 134  
`group_proxy_cls` (in- has\_note(`indico.modules.events.sessions.models.blocks.SessionBlock`  
`dico.modules.events.papers.models.reviews.PaperReview` attribute), 242  
 attribute), 199 has\_photo(`indico.modules.rb.models.rooms.Room` at-  
 tribute), 291  
`GroupProxy` (class in `indico.modules.groups.core`), has\_picture(`indico.modules.users.models.users.User`  
 310 attribute), 273

423

- `id(indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry, attribute), 150` `id(indico.modules.events.models.principals.EventPrincipal, attribute), 131`
- `id(indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate, attribute), 150` `id(indico.modules.events.models.references.EventReference, attribute), 131`
- `id(indico.modules.events.abstracts.models.files.AbstractFile, attribute), 152` `id(indico.modules.events.models.references.ReferenceModelBase, attribute), 132`
- `id(indico.modules.events.abstracts.models.persons.AbstractPersonLink, attribute), 152` `id(indico.modules.events.models.references.ReferenceType, attribute), 132`
- `id(indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion, attribute), 153` `id(indico.modules.events.models.reviews.ProposalGroupProxy, attribute), 133`
- `id(indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating, attribute), 154` `id(indico.modules.events.models.series.EventSeries, attribute), 135`
- `id(indico.modules.events.abstracts.models.reviews.AbstractReview, attribute), 155` `id(indico.modules.events.models.settings.EventSetting, attribute), 135`
- `id(indico.modules.events.abstracts.settings.BOASortField, attribute), 165` `id(indico.modules.events.models.settings.EventSettingPrincipal, attribute), 136`
- `id(indico.modules.events.agreements.models.agreements.Agreement, attribute), 166` `id(indico.modules.events.models.static_list_links.StaticListLink, attribute), 137`
- `id(indico.modules.events.contributions.models.contributions.Contribution, attribute), 170` `id(indico.modules.events.notes.models.notes.EventNote, attribute), 191`
- `id(indico.modules.events.contributions.models.fields.ContributionField, attribute), 173` `id(indico.modules.events.notes.models.notes.EventNoteRevision, attribute), 192`
- `id(indico.modules.events.contributions.models.persons.ContributionPersonLink, attribute), 174` `id(indico.modules.events.papers.models.comments.PaperReviewComment, attribute), 194`
- `id(indico.modules.events.contributions.models.persons.SubContributionPersonLink, attribute), 175` `id(indico.modules.events.papers.models.competences.PaperCompetence, attribute), 195`
- `id(indico.modules.events.contributions.models.principals.ContributionPrincipal, attribute), 176` `id(indico.modules.events.papers.models.files.PaperFile, attribute), 195`
- `id(indico.modules.events.contributions.models.references.ContributionReference, attribute), 177` `id(indico.modules.events.papers.models.review_questions.PaperReviewQuestion, attribute), 197`
- `id(indico.modules.events.contributions.models.references.SubContributionReference, attribute), 177` `id(indico.modules.events.papers.models.review_ratings.PaperReviewRating, attribute), 198`
- `id(indico.modules.events.contributions.models.subcontributions.SubContribution, attribute), 177` `id(indico.modules.events.papers.models.reviews.PaperReview, attribute), 199`
- `id(indico.modules.events.contributions.models.types.ContributionType, attribute), 179` `id(indico.modules.events.papers.models.revisions.PaperRevision, attribute), 201`
- `id(indico.modules.events.layout.models.images.ImageFile, attribute), 182` `id(indico.modules.events.papers.models.templates.PaperTemplate, attribute), 202`
- `id(indico.modules.events.layout.models.menu.EventPage, attribute), 183` `id(indico.modules.events.payment.models.transactions.PaymentTransaction, attribute), 207`
- `id(indico.modules.events.layout.models.menu.MenuEntry, attribute), 183` `id(indico.modules.events.registration.models.form_fields.RegistrationFormField, attribute), 217`
- `id(indico.modules.events.layout.models.menu.TransientMenuEntry, attribute), 185` `id(indico.modules.events.registration.models.form_fields.RegistrationFormField, attribute), 218`
- `id(indico.modules.events.logs.models.entries.EventLogEntry, attribute), 187` `id(indico.modules.events.registration.models.form_fields.RegistrationFormField, attribute), 218`
- `id(indico.modules.events.models.events.Event, attribute), 124` `id(indico.modules.events.registration.models.forms.RegistrationForm, attribute), 220`
- `id(indico.modules.events.models.persons.EventPerson, attribute), 128` `id(indico.modules.events.registration.models.invitations.RegistrationInvitation, attribute), 222`
- `id(indico.modules.events.models.persons.EventPersonLink, attribute), 129` `id(indico.modules.events.registration.models.items.RegistrationFormItem, attribute), 223`
- `id(indico.modules.events.models.persons.PersonLinkBase, attribute), 130` `id(indico.modules.events.registration.models.items.RegistrationFormPerson, attribute), 225`



- `id (indico.modules.events.registration.models.items.RegistrationFormSection attribute)`, 225
- `id (indico.modules.events.registration.models.items.RegistrationFormText attribute)`, 226
- `id (indico.modules.events.registration.models.registrations.Registration attribute)`, 213
- `id (indico.modules.events.reminders.models.reminders.EventReminder attribute)`, 235
- `id (indico.modules.events.requests.models.requests.Request attribute)`, 237
- `id (indico.modules.events.sessions.models.blocks.SessionBlock attribute)`, 242
- `id (indico.modules.events.sessions.models.persons.SessionBlockParticipant attribute)`, 243
- `id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute)`, 243
- `id (indico.modules.events.sessions.models.sessions.Session attribute)`, 240
- `id (indico.modules.events.static.models.static.StaticSite attribute)`, 262
- `id (indico.modules.events.surveys.models.items.SurveyItem attribute)`, 248
- `id (indico.modules.events.surveys.models.items.SurveyQuestion attribute)`, 249
- `id (indico.modules.events.surveys.models.items.SurveySection attribute)`, 249
- `id (indico.modules.events.surveys.models.items.SurveyText attribute)`, 250
- `id (indico.modules.events.surveys.models.submissions.SurveySubmission attribute)`, 251
- `id (indico.modules.events.surveys.models.surveys.Survey attribute)`, 246
- `id (indico.modules.events.timetable.models.breaks.Break attribute)`, 253
- `id (indico.modules.events.timetable.models.entries.TimetableEntry attribute)`, 255
- `id (indico.modules.events.tracks.models.principals.TrackPrincipal attribute)`, 261
- `id (indico.modules.events.tracks.models.tracks.Track attribute)`, 259
- `id (indico.modules.groups.models.groups.LocalGroup attribute)`, 310
- `id (indico.modules.networks.models.networks.IPNetworkGroup attribute)`, 326
- `id (indico.modules.news.models.news.NewsItem attribute)`, 326
- `id (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute)`, 294
- `id (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute)`, 295
- `id (indico.modules.rb.models.blockings.Blocking attribute)`, 294
- `id (indico.modules.rb.models.equipment.EquipmentType attribute)`, 295
- `id (indico.modules.rb.models.locations.Location attribute)`, 295
- `id (indico.modules.rb.models.map_areas.MapArea attribute)`, 296
- `id (indico.modules.rb.models.photos.Photo attribute)`, 296
- `id (indico.modules.rb.models.reservation_edit_logs.ReservationEditLog attribute)`, 300
- `id (indico.modules.rb.models.reservations.Reservation attribute)`, 298
- `id (indico.modules.rb.models.reservations.ReservationLink attribute)`, 299
- `id (indico.modules.rb.models.room_attributes.RoomAttribute attribute)`, 292
- `id (indico.modules.rb.models.rooms.Room attribute)`, 291
- `id (indico.modules.users.models.affiliations.UserAffiliation attribute)`, 276
- `id (indico.modules.users.models.emails.UserEmail attribute)`, 276
- `id (indico.modules.users.models.settings.UserSetting attribute)`, 277
- `id (indico.modules.users.models.users.User attribute)`, 273
- `id (indico.modules.vc.models.vc_rooms.VCRoom attribute)`, 312
- `id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute)`, 313
- `id (indico.modules.users.models.users.ProfilePictureSource attribute)`, 272
- `identifier (indico.modules.auth.models.identities.Identity attribute)`, 302
- `identifier (indico.modules.events.agreements.models.agreements.Agreement attribute)`, 166
- `identifier (indico.modules.events.models.persons.EventPerson attribute)`, 128
- `identifier (indico.modules.events.registration.models.forms.RegistrationForm attribute)`, 220
- `identifier (indico.modules.groups.core.GroupProxy attribute)`, 311
- `identifier (indico.modules.users.models.users.User attribute)`, 273
- `identities (indico.modules.users.models.users.User attribute)`, 273
- `Identity (class in indico.modules.auth.models.identities)`, 302
- `identity_data (indico.modules.auth.models.registration_requests.RegistrationRequest attribute)`, 303
- `IDENTITY_PROVIDERS (built-in variable)`, 48
- `IDPlaceholder (class in indico.modules.events.registration.placeholders.registrations)`, 230
- `IgnoredTransactionAction`, 207

[image\\_created](#) (in module [indico.core.signals.event\\_management](#)), 87  
[image\\_deleted](#) (in module [indico.core.signals.event\\_management](#)), 87  
[ImageFile](#) (class in [indico.modules.events.layout.models.images](#)), 182  
[ImagePreviewer](#) (class in [indico.modules.attachments.preview](#)), 288  
[impersonate\\_user\(\)](#) (in module [indico.modules.auth.util](#)), 303  
[import\\_contributions\\_from\\_csv\(\)](#) (in module [indico.modules.events.contributions.util](#)), 180  
[import\\_registrations\\_from\\_csv\(\)](#) (in module [indico.modules.events.registration.util](#)), 229  
[import\\_tasks](#) (in module [indico.core.signals.core](#)), 84  
[imported](#) (in module [indico.core.signals.event](#)), 85  
[in\\_progress](#) ([indico.modules.events.abstracts.models.abstracts.AbstractEventReviewingState](#) attribute), 147  
[include\\_authors](#) (in [indico.modules.events.abstracts.models.email\\_templates.AbstractEmailTemplate](#) attribute), 151  
[include\\_coauthors](#) (in [indico.modules.events.abstracts.models.email\\_templates.AbstractEmailTemplate](#) attribute), 151  
[include\\_description](#) (in [indico.modules.events.reminders.models.reminders.EventReminder](#) attribute), 235  
[include\\_submitter](#) (in [indico.modules.events.abstracts.models.email\\_templates.AbstractEmailTemplate](#) attribute), 151  
[include\\_summary](#) (in [indico.modules.events.reminders.models.reminders.EventReminder](#) attribute), 235  
[indico.core.oauth](#) (module), 304  
[indico.core.oauth.models.applications](#) (module), 304  
[indico.core.oauth.models.tokens](#) (module), 307  
[indico.core.plugins](#) (module), 77  
[indico.modules.attachments](#) (module), 281  
[indico.modules.attachments.models.attachments](#) (module), 281  
[indico.modules.attachments.models.folders](#) (module), 283  
[indico.modules.attachments.models.principals](#) (module), 285  
[indico.modules.attachments.operations](#) (module), 287  
[indico.modules.attachments.preview](#) (module), 288  
[indico.modules.attachments.util](#) (module), 287  
[indico.modules.auth](#) (module), 302  
[indico.modules.auth.models.identities](#) (module), 302  
[indico.modules.auth.models.registration\\_requests](#) (module), 303  
[indico.modules.auth.util](#) (module), 303  
[indico.modules.categories](#) (module), 263  
[indico.modules.categories.fields](#) (module), 333  
[indico.modules.categories.models.categories](#) (module), 263  
[indico.modules.categories.models.principals](#) (module), 267  
[indico.modules.categories.models.settings](#) (module), 268  
[indico.modules.categories.operations](#) (module), 268  
[indico.modules.categories.serialize](#) (module), 269  
[indico.modules.categories.settings](#) (module), 270  
[indico.modules.categories.util](#) (module), 268  
[indico.modules.designer](#) (module), 316  
[indico.modules.designer.models.images](#) (module), 316  
[indico.modules.designer.models.templates](#) (module), 317  
[indico.modules.designer.pdf](#) (module), 318  
[indico.modules.designer.placeholders](#) (module), 318  
[indico.modules.designer.util](#) (module), 318  
[indico.modules.events](#) (module), 121  
[indico.modules.events.abstracts](#) (module), 144  
[indico.modules.events.abstracts.fields](#) (module), 328  
[indico.modules.events.abstracts.models.abstracts](#) (module), 144  
[indico.modules.events.abstracts.models.call\\_for\\_abstracts](#) (module), 148  
[indico.modules.events.abstracts.models.comments](#) (module), 149  
[indico.modules.events.abstracts.models.email\\_logs](#) (module), 150  
[indico.modules.events.abstracts.models.email\\_templates](#) (module), 150  
[indico.modules.events.abstracts.models.fields](#) (module), 151  
[indico.modules.events.abstracts.models.files](#) (module), 152  
[indico.modules.events.abstracts.models.persons](#) (module), 152

indico.modules.events.abstracts.models.rendered\_modules (module), 153  
 indico.modules.events.abstracts.models.rendered\_modules.events.layout.models.menu (module), 183  
 indico.modules.events.abstracts.models.rendered\_modules.events.layout.util (module), 185  
 indico.modules.events.abstracts.models.rendered\_modules.events.logs (module), 186  
 indico.modules.events.abstracts.models.rendered\_modules.events.logs.models.entries (module), 154  
 indico.modules.events.abstracts.models.rendered\_modules.events.logs.renderers (module), 186  
 indico.modules.events.abstracts.models.rendered\_modules.events.logs.util (module), 188  
 indico.modules.events.abstracts.operations (module), 156  
 indico.modules.events.abstracts.placeholders (module), 159  
 indico.modules.events.abstracts.settings (module), 164  
 indico.modules.events.abstracts.util (module), 157  
 indico.modules.events.agreements (module), 166  
 indico.modules.events.agreements.models.agreements (module), 166  
 indico.modules.events.agreements.placeholders (module), 168  
 indico.modules.events.agreements.util (module), 167  
 indico.modules.events.contributions (module), 168  
 indico.modules.events.contributions.fields (module), 331  
 indico.modules.events.contributions.models.contributions (module), 169  
 indico.modules.events.contributions.models.contributions.fields (module), 172  
 indico.modules.events.contributions.models.contributions.persons (module), 174  
 indico.modules.events.contributions.models.contributions.principals (module), 175  
 indico.modules.events.contributions.models.references (module), 176  
 indico.modules.events.contributions.models.subcontributions (module), 177  
 indico.modules.events.contributions.models.typecontributions (module), 178  
 indico.modules.events.contributions.operations (module), 179  
 indico.modules.events.contributions.util (module), 180  
 indico.modules.events.features (module), 181  
 indico.modules.events.features.util (module), 181  
 indico.modules.events.fields (module), 327  
 indico.modules.events.layout (module), 182  
 indico.modules.events.layout.models.images (module), 182  
 indico.modules.events.layout.models.rendered\_modules (module), 183  
 indico.modules.events.layout.models.rendered\_modules.events.layout.util (module), 185  
 indico.modules.events.layout.models.rendered\_modules.events.logs (module), 186  
 indico.modules.events.layout.models.rendered\_modules.events.logs.models.entries (module), 186  
 indico.modules.events.layout.models.rendered\_modules.events.logs.renderers (module), 188  
 indico.modules.events.layout.models.rendered\_modules.events.logs.util (module), 188  
 indico.modules.events.layout.models.rendered\_modules.events.models.events (module), 121  
 indico.modules.events.layout.models.rendered\_modules.events.models.persons (module), 127  
 indico.modules.events.layout.models.rendered\_modules.events.models.principals (module), 130  
 indico.modules.events.layout.models.rendered\_modules.events.models.references (module), 131  
 indico.modules.events.layout.models.rendered\_modules.events.models.reviews (module), 133  
 indico.modules.events.layout.models.rendered\_modules.events.models.series (module), 135  
 indico.modules.events.layout.models.rendered\_modules.events.models.settings (module), 135  
 indico.modules.events.layout.models.rendered\_modules.events.models.static\_list\_links (module), 136  
 indico.modules.events.layout.models.rendered\_modules.events.notes (module), 190  
 indico.modules.events.layout.models.rendered\_modules.events.notes.models.notes (module), 190  
 indico.modules.events.layout.models.rendered\_modules.events.notes.util (module), 192  
 indico.modules.events.layout.models.rendered\_modules.events.operations (module), 137  
 indico.modules.events.layout.models.rendered\_modules.events.papers (module), 192  
 indico.modules.events.layout.models.rendered\_modules.events.papers.fields (module), 331  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.call\_for\_papers (module), 193  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.comments (module), 194  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.competences (module), 195  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.files (module), 195  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.papers (module), 196  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.review\_questions (module), 197  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.review\_ratings (module), 198  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.reviews (module), 198  
 indico.modules.events.layout.models.rendered\_modules.events.papers.models.revisions (module), 198

(*module*), 200  
indico.modules.events.papers.models.templates (*module*), 202  
indico.modules.events.papers.models.userindico.modules.events.requests.util (*module*), 237  
indico.modules.events.papers.operations (*module*), 203  
indico.modules.events.papers.util (*module*), 204  
indico.modules.events.payment (*module*), 207  
indico.modules.events.payment.models.transactions (*module*), 207  
indico.modules.events.payment.plugins (*module*), 209  
indico.modules.events.payment.util (*module*), 208  
indico.modules.events.persons (*module*), 210  
indico.modules.events.persons.operations (*module*), 210  
indico.modules.events.persons.placeholder (*module*), 210  
indico.modules.events.registration (*module*), 212  
indico.modules.events.registration.models.form (*module*), 217  
indico.modules.events.registration.models.forms (*module*), 219  
indico.modules.events.registration.models.invitations (*module*), 221  
indico.modules.events.registration.models.items (*module*), 223  
indico.modules.events.registration.models.registrations (*module*), 212  
indico.modules.events.registration.placeholder (*module*), 231  
indico.modules.events.registration.placeholder (*module*), 229  
indico.modules.events.registration.settings (*module*), 232  
indico.modules.events.registration.stats (*module*), 233  
indico.modules.events.registration.util (*module*), 227  
indico.modules.events.reminders (*module*), 234  
indico.modules.events.reminders.models.reminders (*module*), 234  
indico.modules.events.reminders.util (*module*), 236  
indico.modules.events.requests (*module*), 236  
indico.modules.events.requests.base (*module*), 237  
indico.modules.events.requests.models.requests (*module*), 236  
indico.modules.events.requests.util (*module*), 237  
indico.modules.events.sessions (*module*), 239  
indico.modules.events.sessions.fields (*module*), 333  
indico.modules.events.sessions.models.blocks (*module*), 241  
indico.modules.events.sessions.models.persons (*module*), 243  
indico.modules.events.sessions.models.principals (*module*), 243  
indico.modules.events.sessions.models.sessions (*module*), 239  
indico.modules.events.sessions.operations (*module*), 244  
indico.modules.events.sessions.util (*module*), 244  
indico.modules.events.settings (*module*), 142, 204  
indico.modules.events.static (*module*), 261  
indico.modules.events.static.models.static (*module*), 262  
indico.modules.events.static.util (*module*), 263  
indico.modules.events.surveys (*module*), 245  
indico.modules.events.surveys.models.items (*module*), 248  
indico.modules.events.surveys.models.submissions (*module*), 250  
indico.modules.events.surveys.models.surveys (*module*), 245  
indico.modules.events.surveys.operations (*module*), 251  
indico.modules.events.surveys.util (*module*), 252  
indico.modules.events.timetable (*module*), 253  
indico.modules.events.timetable.models.breaks (*module*), 253  
indico.modules.events.timetable.models.entries (*module*), 254  
indico.modules.events.timetable.operations (*module*), 256  
indico.modules.events.timetable.reschedule (*module*), 258  
indico.modules.events.timetable.util (*module*), 257  
indico.modules.events.tracks (*module*), 259  
indico.modules.events.tracks.models.principals



(module), 260  
 indico.modules.events.tracks.models.tracks (module), 259  
 indico.modules.events.tracks.operations (module), 261  
 indico.modules.events.util (module), 139  
 indico.modules.groups (module), 310  
 indico.modules.groups.core (module), 310  
 indico.modules.groups.models.groups (module), 310  
 indico.modules.groups.util (module), 311  
 indico.modules.networks (module), 325  
 indico.modules.networks.fields (module), 333  
 indico.modules.networks.models.networks (module), 325  
 indico.modules.networks.util (module), 326  
 indico.modules.news (module), 326  
 indico.modules.news.models.news (module), 326  
 indico.modules.news.util (module), 327  
 indico.modules.rb (module), 289  
 indico.modules.rb.models.blocked\_rooms (module), 294  
 indico.modules.rb.models.blocking\_principals (module), 294  
 indico.modules.rb.models.blockings (module), 293  
 indico.modules.rb.models.equipment (module), 295  
 indico.modules.rb.models.locations (module), 295  
 indico.modules.rb.models.map\_areas (module), 296  
 indico.modules.rb.models.photos (module), 296  
 indico.modules.rb.models.reservation\_edit\_logs (module), 299  
 indico.modules.rb.models.reservation\_occurrences (module), 300  
 indico.modules.rb.models.reservations (module), 296  
 indico.modules.rb.models.room\_attributes (module), 292  
 indico.modules.rb.models.room\_bookable\_hours (module), 293  
 indico.modules.rb.models.room\_nonbookable\_periods (module), 293  
 indico.modules.rb.models.rooms (module), 289  
 indico.modules.rb.models.util (module), 301  
 indico.modules.rb.statistics (module), 302  
 indico.modules.rb.util (module), 301  
 indico.modules.search.base (module), 70  
 indico.modules.search.result\_schemas (module), 71  
 indico.modules.users (module), 271  
 indico.modules.users.ext (module), 280  
 indico.modules.users.models.affiliations (module), 276  
 indico.modules.users.models.emails (module), 276  
 indico.modules.users.models.favorites (module), 276  
 indico.modules.users.models.settings (module), 277  
 indico.modules.users.models.suggestions (module), 276  
 indico.modules.users.models.users (module), 271  
 indico.modules.users.operations (module), 278  
 indico.modules.users.util (module), 279  
 indico.modules.vc (module), 311  
 indico.modules.vc.exceptions (module), 316  
 indico.modules.vc.models.vc\_rooms (module), 312  
 indico.modules.vc.plugins (module), 314  
 indico.modules.vc.util (module), 314  
 indico.web.forms.fields (module), 334  
 IndicoDateField (class in indico.web.forms.fields), 345  
 IndicoDateTimeField (class in indico.web.forms.fields), 340  
 IndicoEmailRecipientsField (class in indico.web.forms.fields), 347  
 IndicoEnumRadioField (class in indico.web.forms.fields), 341  
 IndicoEnumSelectField (class in indico.web.forms.fields), 341  
 IndicoLocationField (class in indico.web.forms.fields), 345  
 IndicoMarkdownField (class in indico.web.forms.fields), 345  
 IndicoPalettePickerField (class in indico.web.forms.fields), 339  
 IndicoPasswordField (class in indico.web.forms.fields), 338  
 IndicoPlugin (class in indico.core.plugins), 77  
 IndicoPluginBlueprint (class in indico.core.plugins), 79  
 IndicoPluginBlueprintSetupState (class in indico.core.plugins), 79  
 IndicoProtectionField (class in indico.web.forms.fields), 345  
 IndicoQuerySelectMultipleCheckboxField (class in indico.web.forms.fields), 345

IndicoQuerySelectMultipleField (class in *indico.web.forms.fields*), 344

IndicoRadioField (class in *indico.web.forms.fields*), 334

IndicoSearchProvider (class in *indico.modules.search.base*), 69, 70

IndicoSelectMultipleCheckboxBooleanField (class in *indico.web.forms.fields*), 346

IndicoSelectMultipleCheckboxField (class in *indico.web.forms.fields*), 334

IndicoSinglePalettePickerField (class in *indico.web.forms.fields*), 339

IndicoStaticTextField (class in *indico.web.forms.fields*), 338

IndicoTagListField (class in *indico.web.forms.fields*), 338

IndicoThemeSelectField (class in *indico.modules.events.fields*), 328

IndicoTimeField (class in *indico.web.forms.fields*), 348

IndicoTimezoneSelectField (class in *indico.web.forms.fields*), 341

IndicoWeekDayRepetitionField (class in *indico.web.forms.fields*), 346

info (*indico.modules.categories.models.categories.EventMessageModel* attribute), 267

info (*indico.modules.rb.models.reservation\_edit\_logs.ReservationEditLog* attribute), 300

inherit\_location (in *indico.modules.events.contributions.models.contributions.Contribution* attribute), 170

inherit\_location (in *indico.modules.events.models.events.Event* attribute), 124

inherit\_location (in *indico.modules.events.sessions.models.blocks.SessionBlock* attribute), 242

inherit\_location (in *indico.modules.events.sessions.models.sessions.Session* attribute), 240

inherit\_location (in *indico.modules.events.timetable.models.breaks.Break* attribute), 253

inheriting\_have\_acl (in *indico.modules.categories.models.categories.Category* attribute), 265

inheriting\_have\_acl (in *indico.modules.events.contributions.models.contributions.Contribution* attribute), 170

inheriting\_have\_acl (in *indico.modules.events.models.events.Event* attribute), 124

inheriting\_have\_acl (in *indico.modules.events.sessions.models.sessions.Session* attribute), 240

attribute), 240

init () (*indico.core.plugins.IndicoPlugin* method), 78

init () (*indico.modules.events.payment.plugins.PaymentPluginMixin* method), 209

init () (*indico.modules.vc.plugins.VCPluginMixin* method), 315

initial\_statuses (in *indico.modules.events.payment.models.transactions.TransactionState* attribute), 208

inject\_bundle (in *module* in *indico.core.signals.plugin*), 88

inject\_bundle () (*indico.core.plugins.IndicoPlugin* method), 78

inject\_vars\_js () (in *indico.core.plugins.IndicoPlugin* method), 78

input\_type (*indico.modules.events.registration.models.form\_fields.RegistrationFormField* attribute), 217

input\_type (*indico.modules.events.registration.models.form\_fields.RegistrationFormField* attribute), 218

input\_type (*indico.modules.events.registration.models.items.RegistrationItem* attribute), 223

input\_type (*indico.modules.events.registration.models.items.RegistrationItem* attribute), 225

input\_type (*indico.modules.events.registration.models.items.RegistrationItem* attribute), 225

input\_type (*indico.modules.events.registration.models.items.RegistrationItem* attribute), 226

insert () (*indico.modules.events.layout.models.menu.MenuEntry* method), 183

internal\_link (in *indico.modules.events.layout.models.menu.MenuEntryType* attribute), 184

introduction (*indico.modules.events.registration.models.forms.RegistrationForm* attribute), 220

introduction (*indico.modules.events.surveys.models.surveys.Survey* attribute), 246

InvalidManualTransactionAction, 207

InvalidTransactionAction, 207

InvalidTransactionStatus, 207

InvitationLinkPlaceholder (class in *indico.modules.events.registration.placeholders.invitations*), 231

invitations (*indico.modules.events.registration.models.forms.RegistrationForm* attribute), 220

InvitationState (class in *indico.modules.events.registration.models.invitations*), 221

invited (*indico.modules.events.abstracts.models.abstracts.AbstractPublication* attribute), 147

invited (*indico.modules.events.abstracts.models.abstracts.AbstractState* attribute), 147

invited\_dt (*indico.modules.events.models.persons.EventPerson* attribute), 128

ip\_network\_group (in- IPNetwork (class in in-  
 dico.modules.attachments.models.principals.AttachmentFolderPrincipal (in-  
 attribute), 286 325  
 ip\_network\_group (in- IPNetworkGroup (class in in-  
 dico.modules.attachments.models.principals.AttachmentPrincipal (in-  
 attribute), 287 325  
 ip\_network\_group (in- is\_accepted (indico.modules.rb.models.reservations.Reservation  
 dico.modules.categories.models.principals.CategoryPrincipal attribute), 298  
 attribute), 267 is\_active (indico.modules.events.contributions.models.fields.Contribution  
 ip\_network\_group (in- attribute), 173  
 dico.modules.events.contributions.models.principals.ContributionPrincipal (in-  
 attribute), 176 attribute), 220  
 ip\_network\_group (in- is\_active (indico.modules.events.registration.models.forms.Registration  
 dico.modules.events.models.principals.EventPrincipal attribute), 213  
 attribute), 131 is\_active (indico.modules.events.surveys.models.surveys.Survey  
 ip\_network\_group (in- attribute), 246  
 dico.modules.events.models.settings.EventSettingPrincipal attribute), 136  
 attribute), 136 is\_active () (indico.modules.users.ext.ExtraUserPreferences  
 class method), 280  
 ip\_network\_group (in- is\_active\_at () (in-  
 dico.modules.events.sessions.models.principals.SessionPrincipal (in-  
 attribute), 244 attribute), 294  
 ip\_network\_group (in- is\_admin (indico.modules.users.models.users.User at-  
 dico.modules.events.tracks.models.principals.TrackPrincipal attribute), 273  
 attribute), 261 is\_always\_visible (in-  
 ip\_network\_group (in- dico.modules.attachments.models.folders.AttachmentFolder  
 attribute), 284  
 attribute), 295 is\_anonymous (indico.modules.events.surveys.models.submissions.Survey  
 ip\_network\_group\_id (in- attribute), 251  
 dico.modules.attachments.models.principals.AttachmentFolderPrincipal (in-  
 attribute), 286 attribute), 298  
 ip\_network\_group\_id (in- is\_author (indico.modules.events.contributions.models.persons.Contribution  
 dico.modules.attachments.models.principals.AttachmentPrincipal attribute), 174  
 attribute), 287 is\_auto\_confirm (in-  
 ip\_network\_group\_id (in- dico.modules.rb.models.rooms.Room attribute),  
 dico.modules.categories.models.principals.CategoryPrincipal 291  
 attribute), 267 is\_blocked (indico.modules.users.models.users.User  
 ip\_network\_group\_id (in- attribute), 273  
 dico.modules.events.contributions.models.principals.ContributionPrincipal (in-  
 attribute), 176 attribute), 298  
 ip\_network\_group\_id (in- method), 298  
 dico.modules.events.models.principals.EventPrincipal booking\_start\_within\_grace\_period()  
 attribute), 131 (in module indico.modules.rb.util), 301  
 ip\_network\_group\_id (in- is\_cancelled (indico.modules.events.registration.models.registrations.Registration  
 dico.modules.events.models.settings.EventSettingPrincipal attribute), 213  
 attribute), 136 is\_cancelled (indico.modules.rb.models.reservation\_occurrences.Reservation  
 ip\_network\_group\_id (in- attribute), 300  
 dico.modules.events.sessions.models.principals.SessionPrincipal (in-  
 attribute), 244 attribute), 298  
 ip\_network\_group\_id (in- is\_clonable (indico.modules.designer.models.templates.DesignerTemplate  
 dico.modules.events.tracks.models.principals.TrackPrincipal attribute), 317  
 attribute), 261 is\_currency\_shown (in-  
 ip\_network\_group\_id (in- dico.modules.events.registration.stats.FieldStats  
 attribute), 234  
 attribute), 295 is\_currency\_shown (in-

[indico.modules.events.registration.stats.StatsBase](#) [is\\_deleted](#) ([indico.modules.rb.models.rooms.Room](#) [attribute](#)), 234  
[is\\_default](#) ([indico.modules.attachments.models.folders.AttachmentFolder](#) [attribute](#)), 284  
[is\\_default](#) ([indico.modules.events.layout.models.menu.EventPage](#) [endant\\_of\(\)](#) ([indico.modules.categories.models.categories.Category](#) [attribute](#)), 183  
[is\\_default](#) ([indico.modules.rb.models.map\\_areas.MapArea](#) [method](#)), 265  
[is\\_deleted](#) ([indico.modules.attachments.models.attachments.Attachment](#) [attribute](#)), 282  
[is\\_deleted](#) ([indico.modules.attachments.models.folders.AttachmentFolder](#) [attribute](#)), 284  
[is\\_deleted](#) ([indico.modules.categories.models.categories.Category](#) [attribute](#)), 265  
[is\\_deleted](#) ([indico.modules.events.abstracts.models.abstracts.Abstract](#) [attribute](#)), 145  
[is\\_deleted](#) ([indico.modules.events.abstracts.models.comments.AbstractComment](#) [attribute](#)), 149  
[is\\_deleted](#) ([indico.modules.events.abstracts.models.review\\_questions.AbstractReviewQuestion](#) [attribute](#)), 153  
[is\\_deleted](#) ([indico.modules.events.contributions.models.contributions.Contribution](#) [attribute](#)), 170  
[is\\_deleted](#) ([indico.modules.events.contributions.models.subcontributions.SubContribution](#) [attribute](#)), 178  
[is\\_deleted](#) ([indico.modules.events.models.events.Event](#) [attribute](#)), 124  
[is\\_deleted](#) ([indico.modules.events.notes.models.notes.EventNote](#) [attribute](#)), 191  
[is\\_deleted](#) ([indico.modules.events.papers.models.comments.PaperReviewComment](#) [attribute](#)), 194  
[is\\_deleted](#) ([indico.modules.events.papers.models.review\\_questions.ProposalReviewQuestion](#) [attribute](#)), 197  
[is\\_deleted](#) ([indico.modules.events.registration.models.form\\_fields.RegistrationFormField](#) [attribute](#)), 217  
[is\\_deleted](#) ([indico.modules.events.registration.models.form\\_fields.RegistrationFormPersonalDataField](#) [attribute](#)), 218  
[is\\_deleted](#) ([indico.modules.events.registration.models.forms.RegistrationForm](#) [attribute](#)), 220  
[is\\_deleted](#) ([indico.modules.events.registration.models.items.RegistrationFormField](#) [attribute](#)), 224  
[is\\_deleted](#) ([indico.modules.events.registration.models.items.RegistrationFormPersonalDataField](#) [attribute](#)), 225  
[is\\_deleted](#) ([indico.modules.events.registration.models.items.RegistrationFormSection](#) [attribute](#)), 225  
[is\\_deleted](#) ([indico.modules.events.registration.models.items.RegistrationFormTitle](#) [attribute](#)), 226  
[is\\_deleted](#) ([indico.modules.events.registration.models.registrations.Registration](#) [attribute](#)), 213  
[is\\_deleted](#) ([indico.modules.events.sessions.models.sessions.Session](#) [final\\_state](#) ([indico.modules.events.abstracts.models.abstracts.Abstract](#) [attribute](#)), 240  
[is\\_deleted](#) ([indico.modules.events.surveys.models.surveys.Survey](#) [attribute](#)), 246  
[is\\_deleted](#) ([indico.modules.rb.models.locations.Location](#) [attribute](#)), 295



`is_in_final_state` (in- `attribute`), 184  
`indico.modules.events.papers.models.papers.Paper` `is_overdue` (`indico.modules.events.reminders.models.reminders.EventReminder` `attribute`), 196  
`is_internal_link` (in- `is_owned_by` () (in- `indico.modules.events.layout.models.menu.MenuEntryMixin` `indico.modules.rb.models.reservations.Reservation` `attribute`), 184 `method`), 298  
`is_judge` () (`indico.modules.events.papers.models.call_for_papers.CallForPapers` `indico.modules.events.layout.models.menu.MenuEntryMixin` `method`), 193 `attribute`), 184  
`is_last_revision` (in- `is_paid` (`indico.modules.events.registration.models.registrations.RegistrationForm` `attribute`), 213 `indico.modules.events.papers.models.revisions.PaperRevision` `attribute`), 201 `is_paper_reviewer` () (in- `indico.modules.events.contributions.models.contributions.ContributionForm` `method`), 170  
`is_link` (`indico.modules.events.layout.models.menu.MenuEntryMixin` `attribute`), 184  
`is_locked` (`indico.modules.events.models.events.Event` `attribute`), 124 `is_parallel` () (in- `indico.modules.events.timetable.models.entries.TimetableEntry` `attribute`), 124  
`is_manager` () (`indico.modules.events.papers.models.call_for_papers.CallForPapers` `method`), 193 `is_participation` (in- `indico.modules.events.registration.models.forms.RegistrationForm` `attribute`), 217  
`is_manager_only` (in- `indico.modules.events.registration.models.form_fields.RegistrationFormFields` `attribute`), 217 `is_pending` (`indico.modules.rb.models.reservations.Reservation` `attribute`), 298  
`is_manager_only` (in- `indico.modules.events.registration.models.form_fields.RegistrationFormFields` `attribute`), 218 `indico.modules.events.registration.models.form_fields.RegistrationFormFields` `attribute`), 273  
`is_manager_only` (in- `is_plugin_link` (in- `indico.modules.events.layout.models.menu.MenuEntryMixin` `attribute`), 224 `attribute`), 184  
`is_manager_only` (in- `is_poster` (`indico.modules.events.sessions.models.sessions.SessionForm` `attribute`), 225 `indico.modules.events.registration.models.items.RegistrationFormFields` `attribute`), 225 `is_previewable` (in- `indico.modules.attachments.models.attachments.AttachmentFile` `attribute`), 226  
`is_manager_only` (in- `is_primary` (`indico.modules.users.models.emails.UserEmail` `attribute`), 276 `indico.modules.events.registration.models.items.RegistrationFormFields` `attribute`), 226  
`is_manual` (`indico.modules.events.payment.models.transactions.PaymentTransaction` `attribute`), 207 `indico.modules.events.contributions.models.subcontributions.SubcontributionForm` `attribute`), 178  
`is_menu_entry_enabled` () (in `module` in- `is_public` (`indico.modules.events.contributions.models.fields.ContributionFormFields` `attribute`), 179 `indico.modules.events.layout.util`), 186 `attribute`), 173  
`is_modification_allowed` () (in- `is_publishable` (in- `indico.modules.events.registration.models.registrations.RegistrationForm` `method`), 220 `attribute`), 213  
`is_modification_open` (in- `is_rejected` (`indico.modules.rb.models.reservation_occurrences.ReservationOccurrence` `attribute`), 300 `indico.modules.events.registration.models.forms.RegistrationFormFields` `attribute`), 220 `is_rejected` (`indico.modules.rb.models.reservations.Reservation` `attribute`), 298  
`is_open` (`indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` `attribute`), 148 `is_relative` (`indico.modules.events.reminders.models.reminders.EventReminder` `attribute`), 235  
`is_open` (`indico.modules.events.papers.models.call_for_papers.CallForPapers` `attribute`), 193 `is_repeating` (`indico.modules.rb.models.reservations.Reservation` `attribute`), 298  
`is_open` (`indico.modules.events.registration.models.forms.RegistrationForm` `attribute`), 220 `is_request_manager` () (in `module` in- `indico.modules.events.requests.util`), 237  
`is_orphan` () (`indico.modules.events.agreements.models.agreements.Agreement` `method`), 166 `is_required` (`indico.modules.events.abstracts.models.review_questions.ReviewQuestionForm` `attribute`), 153  
`is_orphaned` (`indico.modules.events.layout.models.menu.MenuEntryMixin` `attribute`), 153

is_required (indico.modules.events.contributions.models.fields.ContributionField attribute), 173	is_required (indico.modules.events.contributions.models.persons.SubContributionField attribute), 175
is_required (indico.modules.events.papers.models.reviews.questions.PaperReviewQuestions attribute), 197	is_required (indico.modules.events.papers.models.call_for_papers.CallForPapersForm method), 193
is_required (indico.modules.events.registration.models.forms.fields.RegistrationFormFields() (in module indico.modules.events.surveys.util), 252	is_required (indico.modules.events.registration.models.forms.fields.RegistrationFormFields attribute), 217
is_required (indico.modules.events.registration.models.items.personal_data_type.PersonalDataType attribute), 218	is_required (indico.modules.events.contributions.models.persons.ContributionField attribute), 175
is_required (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 223	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 224	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 225	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 226	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 226	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 318	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 248	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 249	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 249	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 250	is_required (indico.modules.events.models.persons.EventPersonLink attribute), 129
is_reservable (indico.modules.rb.models.rooms.Room attribute), 291	is_reservable (indico.modules.rb.models.rooms.Room attribute), 291
is_reviewer() (indico.modules.events.papers.models.call_for_papers.CallForPapersForm method), 193	is_reviewer() (indico.modules.events.papers.models.call_for_papers.CallForPapersForm method), 193
is_revoked() (indico.core.oauth.models.tokens.OAuthToken attribute), 309	is_revoked() (indico.core.oauth.models.tokens.OAuthToken attribute), 309
is_root (indico.modules.categories.models.categories.Category attribute), 265	is_root (indico.modules.categories.models.categories.Category attribute), 265
is_root (indico.modules.events.layout.models.menu.MenuEntry attribute), 183	is_root (indico.modules.events.layout.models.menu.MenuEntry attribute), 183
is_scheduled (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstractsForm attribute), 148	is_scheduled (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstractsForm attribute), 148
is_scheduled (indico.modules.events.contributions.models.contributions.Contribution attribute), 170	is_scheduled (indico.modules.events.contributions.models.contributions.Contribution attribute), 170
is_scheduled (indico.modules.events.registration.models.forms.RegistrationForm attribute), 220	is_scheduled (indico.modules.events.registration.models.forms.RegistrationForm attribute), 220
is_section (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 224	is_section (indico.modules.events.registration.models.items.RegistrationFormFields attribute), 224
is_sent (indico.modules.events.reminders.models.reminders.EventReminder attribute), 235	is_sent (indico.modules.events.reminders.models.reminders.EventReminder attribute), 235
is_separator (indico.modules.events.layout.models.menu.MenuEntry attribute), 184	is_separator (indico.modules.events.layout.models.menu.MenuEntry attribute), 184
is_speaker (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 152	is_speaker (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 152
is_speaker (indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 174	is_speaker (indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 174

`is_user_registered()` (in- `judge` (`indico.modules.events.papers.models.revisions.PaperRevision`  
`dico.modules.events.models.events.Event` `attribute`), 201  
`method`), 124 `judge_abstract()` (in `module` in-  
`is_valid` (`indico.modules.rb.models.reservation_occurrences.ReservationOccurrence`  
`attribute`), 300 157  
`is_visible` (`indico.modules.events.layout.models.menu.MenuEntryAdmin` (in-  
`attribute`), 184 `dico.modules.events.papers.models.call_for_papers.CallForPaperItem`  
`is_visible` (`indico.modules.events.registration.models.items.RegistrationItem`  
`attribute`), 224 `judge_id` (`indico.modules.events.abstracts.models.abstracts.Abstract`  
`is_visible` (`indico.modules.events.surveys.models.surveys.Survey` `attribute`), 145  
`attribute`), 246 `judge_id` (`indico.modules.events.papers.models.revisions.PaperRevision`  
`is_visible_in()` (in- `attribute`), 201  
`dico.modules.events.models.events.Event` `judge_paper()` (in `module` in-  
`class method`), 125 `dico.modules.events.papers.operations`),  
`is_within_cancel_grace_period` (in- 203  
`dico.modules.rb.models.reservation_occurrences.ReservationOccurrence`  
`attribute`), 300 `judges` (`indico.modules.events.abstracts.models.reviews.AbstractComment`  
`attribute`), 155  
`items` (in `module` `indico.core.signals.menu`), 88 `judges` (`indico.modules.events.papers.models.call_for_papers.CallForPaper`  
`items` (`indico.modules.designer.pdf.TplData` `attribute`), `attribute`), 193  
319 `judges` (`indico.modules.events.papers.models.reviews.PaperCommentVisi`  
`items` (`indico.modules.events.surveys.models.surveys.Survey` `attribute`), 199  
`attribute`), 246 `judgment_comment` (in-  
`iter_all_multipass_groups()` (in- `dico.modules.events.abstracts.models.abstracts.Abstract`  
`dico.modules.users.models.users.User` `attribute`), 145  
`method`), 274 `judgment_comment` (in-  
`iter_choices()` (in- `dico.modules.events.papers.models.papers.Paper`  
`dico.web.forms.fields.IndicoEnumSelectField` `attribute`), 196  
`method`), 341 `judgment_comment` (in-  
`iter_choices()` (in- `dico.modules.events.papers.models.revisions.PaperRevision`  
`dico.web.forms.fields.IndicoSelectMultipleCheckboxBooleanField` `attribute`), 201  
`method`), 346 `judgment_dt` (`indico.modules.events.abstracts.models.abstracts.Abstract`  
`iter_create_occurrences()` (in- `attribute`), 146  
`dico.modules.rb.models.reservation_occurrences.ReservationOccurrence`  
`class method`), 300 `attribute`), 201  
`iter_days()` (`indico.modules.events.models.events.Event` `judgment_instructions` (in-  
`method`), 125 `dico.modules.events.abstracts.models.call_for_abstracts.CallForAbstract`  
`attribute`), 148  
`iter_identifiers()` (in- `JudgmentCommentPlaceholder` (class in in-  
`dico.modules.users.models.users.User` `dico.modules.events.abstracts.placeholders`),  
`method`), 274 164  
`iter_param_info()` (in- `ContributionsPlaceholder`  
`dico.modules.events.persons.placeholders.ContributionsPlaceholder`  
`class method`), 210  
`iter_param_info()` (in- `key` (`indico.modules.search.result_schemas.BucketSchema`  
`dico.modules.events.registration.placeholders.registrations.RegistrationPlaceholder`  
`class method`), 230 `key_location` (`indico.modules.rb.models.rooms.Room`  
`attribute`), 291  
`iter_start_time()` (in- `keywords` (`indico.modules.events.contributions.models.contributions.Contributions`  
`dico.modules.rb.models.reservation_occurrences.ReservationOccurrence` `attribute`), 170  
`static method`), 300 `kind` (`indico.modules.events.logs.models.entries.EventLogEntry`  
`attribute`), 187

## J

JSONField (class in `indico.web.forms.fields`), 334

judge (`indico.modules.events.abstracts.models.abstracts.Abstract` `attribute`), 145

## L

		<i>dico.modules.events.persons.placeholders</i> ), 211
label ( <i>indico.modules.events.models.events.Event</i> attribute), 125	LastNamePlaceholder (class in in-	
label ( <i>indico.modules.search.result_schemas.AggregationSchema</i> attribute), 68	<i>dico.modules.events.registration.placeholders.invitations</i> ), 232	
label_id ( <i>indico.modules.events.models.events.Event</i> attribute), 125	LastNamePlaceholder (class in in-	
label_message (in-	<i>dico.modules.events.registration.placeholders.registrations</i> ), 230	
<i>dico.modules.events.models.events.Event</i> attribute), 125	latest_date ( <i>indico.web.forms.fields.IndicoDateField</i> attribute), 345	
last_f ( <i>indico.modules.users.models.users.NameFormat</i> attribute), 271	latest_dt ( <i>indico.web.forms.fields.IndicoDateTimeField</i> attribute), 340	
last_f_upper ( <i>indico.modules.users.models.users.NameFormat</i> attribute), 271	latitude ( <i>indico.modules.rb.models.rooms.Room</i> attribute), 291	
last_first ( <i>indico.modules.users.models.users.NameFormat</i> attribute), 271	layout ( <i>indico.modules.events.papers.models.reviews.PaperReviewType</i> attribute), 200	
last_first_upper (in-	layout_review_questions (in-	
<i>dico.modules.users.models.users.NameFormat</i> attribute), 271	<i>dico.modules.events.papers.models.call_for_papers.CallForPaper</i> attribute), 193	
last_login_dt (in-	layout_reviewer_deadline (in-	
<i>dico.modules.auth.models.identities.Identity</i> attribute), 302	<i>dico.modules.events.papers.models.call_for_papers.CallForPaper</i> attribute), 193	
last_login_dt (in-	layout_reviewer_deadline_enforced (in-	
<i>dico.modules.users.models.users.User</i> attribute), 274	<i>dico.modules.events.papers.models.call_for_papers.CallForPaper</i> attribute), 193	
last_login_ip (in-	layout_reviewers (in-	
<i>dico.modules.auth.models.identities.Identity</i> attribute), 302	<i>dico.modules.events.papers.models.call_for_papers.CallForPaper</i> attribute), 193	
last_name ( <i>indico.modules.events.models.persons.EventPerson</i> attribute), 128	layout_reviewing_enabled (in-	
last_name ( <i>indico.modules.events.models.persons.PersonLinkBase</i> attribute), 130	<i>dico.modules.events.papers.models.call_for_papers.CallForPaper</i> attribute), 193	
last_name ( <i>indico.modules.events.registration.models.invitations.RegistrationInvitation</i> attribute), 222	lecture ( <i>indico.modules.events.models.events.EventType</i> attribute), 127	
last_name ( <i>indico.modules.events.registration.models.items.PersonalDataType</i> attribute), 223	legacy_id ( <i>indico.modules.events.contributions.models.fields.Contribution</i> attribute), 173	
last_name ( <i>indico.modules.events.registration.models.registrations.Registration</i> attribute), 214	legacy_name ( <i>indico.modules.events.models.events.EventType</i> attribute), 127	
last_name ( <i>indico.modules.users.models.users.User</i> attribute), 274	limit_reached (in-	
last_revision (in-	<i>dico.modules.events.registration.models.forms.RegistrationForm</i> attribute), 220	
<i>dico.modules.events.papers.models.papers.Paper</i> attribute), 196	limit_reached (in-	
last_used_dt ( <i>indico.core.oauth.models.tokens.OAuthToken</i> attribute), 309	<i>dico.modules.events.surveys.models.surveys.Survey</i> attribute), 246	
last_used_dt ( <i>indico.core.oauth.models.tokens.TokenModelBase</i> attribute), 310	limit_reached (in-	
last_used_dt ( <i>indico.modules.events.models.static_list_links.StaticListLink</i> attribute), 137	<i>dico.modules.events.surveys.models.surveys.SurveyState</i> attribute), 247	
last_used_ip ( <i>indico.core.oauth.models.tokens.OAuthToken</i> attribute), 309	link ( <i>indico.modules.attachments.models.attachments.AttachmentType</i> attribute), 283	
last_used_ip ( <i>indico.core.oauth.models.tokens.TokenModelBase</i> attribute), 310	link ( <i>indico.modules.rb.models.reservations.Reservation</i> attribute), 298	
LastNamePlaceholder (class in in-	link_backref_lazy (in-	
	<i>dico.modules.attachments.models.folders.AttachmentFolder</i> attribute), 285	
	link_backref_name (in-	



[dico.modules.attachments.models.folders.AttachmentFolder](#) attribute), 299  
[attribute](#)), 285 [linked\\_event\\_id](#) (in-  
[link\\_backref\\_name](#) (in- [dico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#)  
[dico.modules.events.notes.models.notes.EventNote](#) attribute), 313  
[attribute](#)), 191 [linked\\_field](#) ([indico.web.forms.fields.IndicoDateField](#)  
[link\\_backref\\_name](#) (in- attribute), 345  
[dico.modules.rb.models.reservations.ReservationLink](#) [linked\\_field](#) ([indico.web.forms.fields.IndicoDateTimeField](#)  
[attribute](#)), 299 attribute), 340  
[link\\_id](#) ([indico.modules.rb.models.reservations.ReservationLink](#) [linked\\_object](#) (in-  
[attribute](#)), 298 [dico.modules.rb.models.reservations.Reservation](#)  
[link\\_object](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#)), 298  
[attribute](#)), 313 [linked\\_object\\_attr](#) (in-  
[link\\_type](#) ([indico.modules.attachments.models.folders.AttachmentFolder](#) [indico.modules.events.abstracts.fields.AbstractPersonLinkListField](#)  
[attribute](#)), 285 attribute), 329  
[link\\_type](#) ([indico.modules.events.notes.models.notes.EventNote](#) [linked\\_object\\_attr](#) (in-  
[attribute](#)), 191 [dico.modules.events.contributions.fields.ContributionPersonLinkListField](#)  
[link\\_type](#) ([indico.modules.rb.models.reservations.ReservationLink](#) attribute), 331  
[attribute](#)), 299 [linked\\_object\\_attr](#) (in-  
[link\\_type](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#) [indico.modules.events.contributions.fields.SubContributionPersonLinkListField](#)  
[attribute](#)), 313 attribute), 331  
[link\\_url](#) ([indico.modules.attachments.models.attachments.Attachment](#) [linked\\_object\\_attr](#) (in-  
[attribute](#)), 282 [dico.modules.events.fields.EventPersonLinkListField](#)  
[link\\_url](#) ([indico.modules.events.layout.models.menu.MenuEntry](#) attribute), 327  
[attribute](#)), 183 [linked\\_object\\_attr](#) (in-  
[link\\_user\\_by\\_email\(\)](#) (in- [dico.modules.events.fields.PersonLinkListFieldBase](#)  
[dico.modules.events.models.persons.EventPerson](#) attribute), 328  
[class method](#)), 128 [linked\\_object\\_attr](#) (in-  
[linked\\_block](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#) [indico.modules.events.sessions.fields.SessionBlockPersonLinkListField](#)  
[attribute](#)), 313 attribute), 333  
[linked\\_contrib](#) (in- [LinkPlaceholder](#) (class in in-  
[dico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#) [dico.modules.events.registration.placeholders.registrations](#)),  
[attribute](#)), 313 231  
[linked\\_date\\_validator](#) (in- [list\\_link\\_type](#) (in-  
[dico.web.forms.fields.IndicoDateField](#) at- [dico.modules.events.util.ListGeneratorBase](#)  
[tribute](#)), 345 attribute), 139  
[linked\\_datetime\\_validator](#) (in- [ListGeneratorBase](#) (class in in-  
[dico.web.forms.fields.IndicoDateTimeField](#) [dico.modules.events.util](#)), 139  
[attribute](#)), 340 [load\(\)](#) ([indico.modules.events.models.static\\_list\\_links.StaticListLink](#)  
[linked\\_event](#) ([indico.modules.attachments.models.folders.AttachmentFolder](#) [class method](#)), 137  
[attribute](#)), 285 [load\(\)](#) ([indico.modules.users.ext.ExtraUserPreferences](#)  
[linked\\_event](#) ([indico.modules.events.notes.models.notes.EventNote](#) [method](#)), 281  
[attribute](#)), 191 [load\\_identity\\_info\(\)](#) (in module in-  
[linked\\_event](#) ([indico.modules.rb.models.reservations.ReservationLink](#) [dico.modules.auth.util](#)), 303  
[attribute](#)), 299 [local\\_group](#) ([indico.modules.attachments.models.principals.AttachmentPrincipal](#)  
[linked\\_event](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#) [attribute](#)), 286  
[attribute](#)), 313 [local\\_group](#) ([indico.modules.attachments.models.principals.AttachmentPrincipal](#)  
[linked\\_event\\_id](#) (in- attribute), 287  
[dico.modules.attachments.models.folders.AttachmentFolder](#) [local\\_group](#) ([indico.modules.categories.models.principals.CategoryPrincipal](#)  
[attribute](#)), 285 attribute), 267  
[linked\\_event\\_id](#) (in- [local\\_group](#) ([indico.modules.events.contributions.models.principals.ContributionPrincipal](#)  
[dico.modules.events.notes.models.notes.EventNote](#) attribute), 176  
[attribute](#)), 191 [local\\_group](#) ([indico.modules.events.models.principals.EventPrincipal](#)  
[linked\\_event\\_id](#) (in- attribute), 131  
[dico.modules.rb.models.reservations.ReservationLink](#) [local\\_group](#) ([indico.modules.events.models.settings.EventSettingPrincipal](#)

- [attribute\), 136](#)
- [local\\_group \(indico.modules.events.sessions.models.principals.SessionPrincipal attribute\), 244](#)
- [local\\_group \(indico.modules.events.tracks.models.principals.TrackPrincipal attribute\), 261](#)
- [local\\_group \(indico.modules.rb.models.blocking\\_principals.BlockingPrincipal attribute\), 295](#)
- [local\\_group\\_id \(in- dico.modules.attachments.models.principals.AttachmentFolderPrincipal attribute\), 286](#)
- [local\\_group\\_id \(in- dico.modules.attachments.models.principals.AttachmentPrincipal attribute\), 287](#)
- [local\\_group\\_id \(in- dico.modules.categories.models.principals.CategoryPrincipal attribute\), 267](#)
- [local\\_group\\_id \(in- dico.modules.events.contributions.models.principals.ContributionPrincipal attribute\), 176](#)
- [local\\_group\\_id \(in- dico.modules.events.models.principals.EventPrincipal attribute\), 131](#)
- [local\\_group\\_id \(in- dico.modules.events.models.settings.EventSettingPrincipal attribute\), 136](#)
- [local\\_group\\_id \(in- dico.modules.events.sessions.models.principals.SessionPrincipal attribute\), 244](#)
- [local\\_group\\_id \(in- dico.modules.events.tracks.models.principals.TrackPrincipal attribute\), 261](#)
- [local\\_group\\_id \(in- dico.modules.rb.models.blocking\\_principals.BlockingPrincipal attribute\), 295](#)
- [LOCAL\\_GROUPS \(built-in variable\), 47](#)
- [LOCAL\\_IDENTITIES \(built-in variable\), 47](#)
- [local\\_identities \(in- dico.modules.users.models.users.User attribute\), 274](#)
- [local\\_identity \(in- dico.modules.users.models.users.User attribute\), 274](#)
- [LOCAL\\_MODERATION \(built-in variable\), 48](#)
- [LOCAL\\_REGISTRATION \(built-in variable\), 48](#)
- [LocalGroup \(class in in- dico.modules.groups.models.groups\), 310](#)
- [localized\\_title \(in- dico.modules.events.layout.models.menu.MenuEntryPrincipal attribute\), 184](#)
- [Location \(class in in- dico.modules.rb.models.locations\), 295](#)
- [location \(indico.modules.rb.models.rooms.Room attribute\), 291](#)
- [location \(indico.modules.search.result\\_schemas.ContributionResultSchema attribute\), 72](#)
- [location \(in- dico.modules.search.result\\_schemas.EventResultSchema attribute\), 71](#)
- [location\\_backref\\_name \(in- dico.modules.events.contributions.models.contributions.Contribution attribute\), 170](#)
- [location\\_backref\\_name \(in- dico.modules.events.models.events.Event attribute\), 125](#)
- [location\\_backref\\_name \(in- dico.modules.events.sessions.models.blocks.SessionBlock attribute\), 242](#)
- [location\\_backref\\_name \(in- dico.modules.events.sessions.models.sessions.Session attribute\), 240](#)
- [location\\_backref\\_name \(in- dico.modules.events.timetable.models.breaks.Break attribute\), 253](#)
- [location\\_changed \(in module in- dico.core.signals.event\), 85](#)
- [location\\_id \(indico.modules.rb.models.rooms.Room attribute\), 291](#)
- [location\\_name \(in- dico.modules.rb.models.reservations.Reservation attribute\), 298](#)
- [location\\_name \(in- indico.modules.rb.models.rooms.Room attribute\), 291](#)
- [location\\_parent \(in- dico.modules.events.contributions.models.contributions.Contribution attribute\), 171](#)
- [location\\_parent \(in- indico.modules.events.contributions.models.subcontributions.SubContribution attribute\), 178](#)
- [location\\_parent \(in- dico.modules.events.sessions.models.blocks.SessionBlock attribute\), 242](#)
- [location\\_parent \(in- dico.modules.events.sessions.models.sessions.Session attribute\), 240](#)
- [location\\_parent \(in- dico.modules.events.timetable.models.breaks.Break attribute\), 253](#)
- [LocationResultSchema \(class in in- dico.modules.search.result\\_schemas\), 73](#)
- [locator \(indico.core.oauth.models.applications.OAuthApplication attribute\), 307](#)
- [locator \(indico.core.oauth.models.tokens.TokenModelBase attribute\), 310](#)
- [locator \(indico.modules.attachments.models.attachments.Attachment attribute\), 282](#)
- [locator \(indico.modules.attachments.models.folders.AttachmentFolder attribute\), 285](#)
- [location \(indico.modules.search.result\\_schemas.ContributionResultSchema attribute\), 72](#)

- [attribute\), 302](#)
- [locator \(indico.modules.auth.models.registration\\_requests.RegistrationRequestModule.events.papers.models.papers.Paper attribute\), 303](#)
- [locator \(indico.modules.categories.models.categories.Categories attribute\), 265](#)
- [locator \(indico.modules.designer.models.images.DesignerImageFile\(indico.modules.events.papers.models.reviews.PaperReview attribute\), 317](#)
- [locator \(indico.modules.designer.models.templates.DesignerTemplate\(indico.modules.events.papers.models.reviews.PaperTypeProxy attribute\), 318](#)
- [locator \(indico.modules.events.abstracts.models.abstracts.Abstract\(indico.modules.events.papers.models.revisions.PaperRevision attribute\), 146](#)
- [locator \(indico.modules.events.abstracts.models.comments.AbstractComment\(indico.modules.events.papers.models.templates.PaperTemplate attribute\), 149](#)
- [locator \(indico.modules.events.abstracts.models.email\\_templates.AK\(indico.modules.events.registration.models.form\\_fields.RegistrationForm attribute\), 151](#)
- [locator \(indico.modules.events.abstracts.models.files.AbstractFile\(indico.modules.events.registration.models.forms.RegistrationForm attribute\), 152](#)
- [locator \(indico.modules.events.abstracts.models.persons.AbstractPerson\(indico.modules.events.registration.models.invitations.RegistrationForm attribute\), 153](#)
- [locator \(indico.modules.events.abstracts.models.review\\_questions\(AbstractReviewQuestion\(indico.modules.events.registration.models.items.RegistrationForm attribute\), 153](#)
- [locator \(indico.modules.events.abstracts.models.reviews.AbstractReview\(indico.modules.events.registration.models.items.RegistrationForm attribute\), 155](#)
- [locator \(indico.modules.events.agreements.models.agreements.Agreement\(indico.modules.events.registration.models.registrations.RegistrationForm attribute\), 166](#)
- [locator \(indico.modules.events.contributions.models.contributions.Contribution\(indico.modules.events.registration.models.registrations.RegistrationForm attribute\), 171](#)
- [locator \(indico.modules.events.contributions.models.fields.ContributionField\(indico.modules.events.reminders.models.reminders.EventReminder attribute\), 173](#)
- [locator \(indico.modules.events.contributions.models.persons.ContributionPerson\(indico.modules.events.requests.models.requests.Request attribute\), 175](#)
- [locator \(indico.modules.events.contributions.models.subcontributions\(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 178](#)
- [locator \(indico.modules.events.contributions.models.types.ContributionType\(indico.modules.events.sessions.models.sessions.Session attribute\), 179](#)
- [locator \(indico.modules.events.layout.models.images.ImageFile\(indico.modules.events.static.models.static.StaticSite attribute\), 182](#)
- [locator \(indico.modules.events.layout.models.menu.EventPage\(indico.modules.events.surveys.models.items.SurveyQuestion attribute\), 183](#)
- [locator \(indico.modules.events.layout.models.menu.MenuEntryMix\(indico.modules.events.surveys.models.items.SurveySection attribute\), 184](#)
- [locator \(indico.modules.events.models.events.Event\(indico.modules.events.surveys.models.items.SurveyText attribute\), 125](#)
- [locator \(indico.modules.events.models.persons.EventPerson\(indico.modules.events.surveys.models.submissions.SurveySubmission attribute\), 128](#)
- [locator \(indico.modules.events.models.references.ReferenceType\(indico.modules.events.surveys.models.surveys.Survey attribute\), 132](#)
- [locator \(indico.modules.events.models.reviews.ProposalGroupProxy\(indico.modules.events.timetable.models.breaks.Break attribute\), 133](#)
- [locator \(indico.modules.events.notes.models.notes.EventNote\(indico.modules.events.timetable.models.entries.TimetableEntry attribute\), 191](#)
- [locator \(indico.modules.events.papers.models.comments.PaperReviewComment\(indico.modules.events.tracks.models.tracks.Track attribute\), 194](#)
- [locator \(indico.modules.events.papers.models.files.PaperFile\(indico.modules.networks.models.networks.IPNetworkGroup attribute\), 195](#)
- [locator \(indico.modules.events.papers.models.papers.Paper attribute\), 196](#)
- [locator \(indico.modules.events.papers.models.review\\_questions.PaperReviewQuestion attribute\), 197](#)
- [locator \(indico.modules.events.papers.models.reviews.PaperReview attribute\), 199](#)
- [locator \(indico.modules.events.papers.models.reviews.PaperTypeProxy attribute\), 200](#)
- [locator \(indico.modules.events.papers.models.revisions.PaperRevision attribute\), 201](#)
- [locator \(indico.modules.events.papers.models.templates.PaperTemplate attribute\), 202](#)
- [locator \(indico.modules.events.registration.models.form\\_fields.RegistrationForm attribute\), 217](#)
- [locator \(indico.modules.events.registration.models.forms.RegistrationForm attribute\), 220](#)
- [locator \(indico.modules.events.registration.models.invitations.RegistrationForm attribute\), 222](#)
- [locator \(indico.modules.events.registration.models.items.RegistrationForm attribute\), 226](#)
- [locator \(indico.modules.events.registration.models.items.RegistrationForm attribute\), 226](#)
- [locator \(indico.modules.events.registration.models.registrations.RegistrationForm attribute\), 214](#)
- [locator \(indico.modules.events.registration.models.registrations.RegistrationForm attribute\), 216](#)
- [locator \(indico.modules.events.reminders.models.reminders.EventReminder attribute\), 235](#)
- [locator \(indico.modules.events.requests.models.requests.Request attribute\), 237](#)
- [locator \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 242](#)
- [locator \(indico.modules.events.sessions.models.sessions.Session attribute\), 240](#)
- [locator \(indico.modules.events.static.models.static.StaticSite attribute\), 262](#)
- [locator \(indico.modules.events.surveys.models.items.SurveyQuestion attribute\), 249](#)
- [locator \(indico.modules.events.surveys.models.items.SurveySection attribute\), 249](#)
- [locator \(indico.modules.events.surveys.models.items.SurveyText attribute\), 250](#)
- [locator \(indico.modules.events.surveys.models.submissions.SurveySubmission attribute\), 251](#)
- [locator \(indico.modules.events.surveys.models.surveys.Survey attribute\), 246](#)
- [locator \(indico.modules.events.timetable.models.breaks.Break attribute\), 253](#)
- [locator \(indico.modules.events.timetable.models.entries.TimetableEntry attribute\), 255](#)
- [locator \(indico.modules.events.tracks.models.tracks.Track attribute\), 260](#)
- [locator \(indico.modules.networks.models.networks.IPNetworkGroup attribute\), 260](#)

- attribute), 326
- locator (*indico.modules.news.models.news.NewsItem* attribute), 326
- locator (*indico.modules.users.models.users.User* attribute), 274
- locator (*indico.modules.vc.models.vc\_rooms.VCRoom* attribute), 312
- locator (*indico.modules.vc.models.vc\_rooms.VCRoomEventAssociation* attribute), 313
- lock\_event() (in module *indico.modules.events.operations*), 138
- log() (*indico.modules.events.abstracts.models.abstracts.Abstract* method), 146
- log() (*indico.modules.events.contributions.models.contributions.Contribution* method), 171
- log() (*indico.modules.events.models.events.Event* method), 125
- log() (*indico.modules.events.registration.models.registrations.Registration* method), 214
- LOG\_DIR (built-in variable), 53
- log\_entries (*indico.modules.events.models.events.Event* attribute), 125
- logged\_date (*indico.modules.events.logs.models.entries.EventLogEntry* attribute), 187
- logged\_dt (*indico.modules.events.logs.models.entries.EventLogEntry* attribute), 187
- logged\_in (in module *indico.core.signals.users*), 90
- LOGGING\_CONFIG\_FILE (built-in variable), 54
- logging\_disabled (in *indico.modules.events.models.events.Event* attribute), 125
- logo (*indico.modules.categories.models.categories.Category* attribute), 265
- logo (*indico.modules.events.models.events.Event* attribute), 125
- logo\_metadata (in *indico.modules.categories.models.categories.Category* attribute), 265
- logo\_metadata (in *indico.modules.events.models.events.Event* attribute), 125
- LOGO\_URL (built-in variable), 51
- logo\_url (*indico.modules.categories.models.categories.Category* attribute), 265
- logo\_url (*indico.modules.events.models.events.Event* attribute), 125
- logo\_url (*indico.modules.events.payment.plugins.PaymentPluginMixin* attribute), 209
- logo\_url (*indico.modules.vc.plugins.VCPluginMixin* attribute), 315
- longitude (*indico.modules.rb.models.rooms.Room* attribute), 291
- ## M
- magnitudes (*indico.web.forms.fields.RelativeDeltaField* attribute), 346
- magnitudes (*indico.web.forms.fields.TimeDeltaField* attribute), 340
- make\_abstract\_form() (in module *indico.modules.events.abstracts.util*), 158
- make\_contribution\_form() (in module *indico.modules.events.contributions.util*), 180
- make\_diff\_log() (in module *indico.modules.events.logs.util*), 188
- make\_email\_primary() (in *indico.modules.users.models.users.User* attribute), 274
- make\_registration\_form() (in module *indico.modules.events.registration.util*), 229
- make\_reminder\_email() (in module *indico.modules.events.reminders.util*), 236
- make\_setup\_state() (in *indico.core.plugins.IndicoPluginBlueprint* method), 79
- make\_survey\_form() (in module *indico.modules.events.surveys.util*), 252
- management (*indico.modules.events.logs.models.entries.EventLogRealm* attribute), 188
- management\_url (in module *indico.core.signals.event\_management*), 87
- manager\_form (*indico.modules.events.requests.base.RequestDefinitionBase* attribute), 238
- manager\_notification\_recipients (in *indico.modules.events.registration.models.forms.RegistrationForm* attribute), 220
- manager\_notifications\_enabled (in *indico.modules.events.registration.models.forms.RegistrationForm* attribute), 220
- manager\_save() (in *indico.modules.events.requests.base.RequestDefinitionBase* class method), 238
- managers (*indico.modules.events.papers.models.call\_for\_papers.CallForPapers* attribute), 193
- managers\_and\_submitters (in *indico.modules.events.contributions.models.fields.ContributionField* attribute), 174
- managers\_only (in *indico.modules.events.contributions.models.fields.ContributionField* attribute), 174
- map\_url (*indico.modules.events.models.events.Event* attribute), 126
- map\_url (*indico.modules.rb.models.rooms.Room* attribute), 291
- map\_url\_template (in *indico.modules.rb.models.locations.Location* attribute), 296
- MapArea (class in in-



[dico.modules.rb.models.map\\_areas](#)), 296  
[mapping](#) ([indico.modules.rb.models.reservations.RepeatMapping](#) attribute), 297  
[mark\\_as\\_duplicate](#) ([indico.modules.events.abstracs.models.reviews.AbstractAction](#) attribute), 154  
[MarkdownPreviewer](#) (class in [indico.modules.attachments.preview](#)), 288  
[marshmallow\\_aliases](#) ([indico.modules.events.abstracs.models.abstracs.Abstract](#) attribute), 146  
[marshmallow\\_aliases](#) ([indico.modules.events.abstracs.models.comments.AbstractComment](#) attribute), 149  
[marshmallow\\_aliases](#) ([indico.modules.events.abstracs.models.reviews.AbstractReview](#) attribute), 156  
[max\\_advance\\_days](#) ([indico.modules.rb.models.rooms.Room](#) attribute), 291  
[MAX\\_UPLOAD\\_FILE\\_SIZE](#) (built-in variable), 56  
[MAX\\_UPLOAD\\_FILES\\_TOTAL\\_SIZE](#) (built-in variable), 56  
[md5](#) ([indico.modules.attachments.models.attachments.AttachmentFile](#) attribute), 283  
[md5](#) ([indico.modules.designer.models.images.DesignerImageFile](#) attribute), 317  
[md5](#) ([indico.modules.events.abstracs.models.files.AbstractFile](#) attribute), 152  
[md5](#) ([indico.modules.events.layout.models.images.ImageFile](#) attribute), 182  
[md5](#) ([indico.modules.events.papers.models.files.PaperFile](#) attribute), 196  
[md5](#) ([indico.modules.events.papers.models.templates.PaperTemplate](#) attribute), 203  
[md5](#) ([indico.modules.events.registration.models.registrations.RegistrationData](#) attribute), 216  
[md5](#) ([indico.modules.events.static.models.static.StaticSite](#) attribute), 262  
[meeting](#) ([indico.modules.events.models.events.EventType](#) attribute), 127  
[members](#) ([indico.modules.groups.models.groups.LocalGroup](#) attribute), 310  
[MEMCACHED\\_SERVERS](#) (built-in variable), 49  
[menu\\_entries\\_for\\_event\(\)](#) (in module [indico.modules.events.layout.util](#)), 186  
[MenuEntry](#) (class in [indico.modules.events.layout.models.menu](#)), 183  
[MenuEntryData](#) (class in [indico.modules.events.layout.util](#)), 185  
[MenuEntryMixin](#) (class in [indico.modules.events.layout.models.menu](#)), 184  
[MenuEntryType](#) (class in [indico.modules.events.layout.models.menu](#)), 184  
[merge](#) ([indico.modules.events.abstracs.models.reviews.AbstractAction](#) attribute), 154  
[merge\\_person\\_info\(\)](#) ([indico.modules.events.models.persons.EventPerson](#) method), 129  
[merge\\_users\(\)](#) (in module [indico.modules.users.util](#)), 279  
[merge\\_users\(\)](#) ([indico.modules.events.models.persons.EventPerson](#) class method), 129  
[merge\\_users\(\)](#) ([indico.modules.events.papers.models.competences.PaperCompetence](#) class method), 195  
[merge\\_users\(\)](#) ([indico.modules.events.registration.models.registrations.Registration](#) class method), 214  
[merge\\_users\(\)](#) ([indico.modules.events.settings.EventACLProxy](#) method), 142, 205  
[merge\\_users\(\)](#) ([indico.modules.users.models.suggestions.SuggestedCategory](#) class method), 277  
[merged](#) (in module [indico.core.signals.users](#)), 90  
[merged](#) ([indico.modules.events.abstracs.models.abstracs.AbstractPublic](#) attribute), 147  
[merged](#) ([indico.modules.events.abstracs.models.abstracs.AbstractState](#) attribute), 147  
[merged\\_into](#) ([indico.modules.events.abstracs.models.abstracs.Abstract](#) attribute), 146  
[merged\\_into\\_id](#) ([indico.modules.events.abstracs.models.abstracs.Abstract](#) attribute), 146  
[merged\\_into\\_user](#) ([indico.modules.users.models.users.User](#) attribute), 274  
[message](#) ([indico.modules.events.reminders.models.reminders.EventReminder](#) attribute), 235  
[message\\_complete](#) ([indico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 220  
[message\\_pending](#) ([indico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 220  
[message\\_unpaid](#) ([indico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 220  
[meta](#) ([indico.modules.events.logs.models.entries.EventLogEntry](#) attribute), 187

[metadata\\_postprocess](#) (in module [indico.core.signals.event](#)), 85  
[modified\\_dt](#) ([indico.modules.search.result\\_schemas.AttachmentResultSchema](#) attribute), 72  
[mgmt\\_field](#) ([indico.modules.events.contributions.models.fields.ContributionField](#) attribute), 73  
[mixed](#) ([indico.modules.events.abstracts.models.abstracts.AbstractReviewingSchema](#) attribute), 147  
[modification\\_enabled](#) (in module [indico.modules.rb.models.reservations.ReservationForm](#)), 298  
[modification\\_end\\_dt](#) (in module [indico.modules.events.registration.models.forms.RegistrationForm](#)), 220  
[modification\\_end\\_dt](#) ([indico.modules.events.registration.util](#)), 229  
[modification\\_end\\_dt](#) ([indico.modules.events.abstracts.models.call\\_for\\_abstracts.CallForAbstracts](#) attribute), 148  
[modification\\_end\\_dt](#) ([indico.modules.categories.models.settings.CategorySetting](#) attribute), 268  
[modification\\_end\\_dt](#) (in module [indico.modules.events.logs.models.entries.EventLogEntry](#) attribute), 187  
[modification\\_ended](#) (in module [indico.modules.events.models.settings.EventSetting](#) attribute), 135  
[modification\\_ended](#) ([indico.modules.events.abstracts.models.abstracts.AbstractEvent](#) attribute), 146  
[modification\\_ended](#) ([indico.modules.events.models.settings.EventSettingPrincipal](#) attribute), 136  
[modification\\_ended](#) (in module [indico.modules.users.models.settings.UserSetting](#) attribute), 273  
[modification\\_ended](#) ([indico.modules.events.abstracts.models.call\\_for\\_abstracts.CallForAbstracts](#) attribute), 148  
[modification\\_mode](#) (in module [indico.modules.rb.models.reservations.RepeatFrequency](#) attribute), 296  
[modification\\_mode](#) ([indico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 220  
[ModificationMode](#) (class in [indico.modules.categories.models.categories.Category](#) method), 266  
[ModificationMode](#) (class in [indico.modules.events.registration.models.forms](#)), 219  
[move\(\)](#) ([indico.modules.events.layout.models.menu.MenuEntry](#) method), 183  
[move\(\)](#) ([indico.modules.events.models.events.Event](#) method), 126  
[move\(\)](#) ([indico.modules.events.timetable.models.entries.TimetableEntry](#) method), 255  
[modified\\_by](#) ([indico.modules.events.abstracts.models.abstracts.AbstractComment](#) attribute), 149  
[modified\\_by](#) ([indico.modules.events.abstracts.models.comments.AbstractComment](#) attribute), 149  
[modified\\_by](#) ([indico.modules.events.papers.models.comments.PaperReviewComment](#) attribute), 194  
[modified\\_by](#) ([indico.modules.categories.operations](#)), 268  
[modified\\_by\\_id](#) (in module [indico.modules.events.timetable.models.entries.TimetableEntry](#) attribute), 146  
[modified\\_by\\_id](#) ([indico.modules.events.abstracts.models.abstracts.AbstractComment](#) attribute), 146  
[modified\\_by\\_id](#) ([indico.modules.events.models.events.Event](#) attribute), 149  
[modified\\_by\\_id](#) ([indico.modules.events.abstracts.models.comments.AbstractComment](#) attribute), 149  
[modified\\_by\\_id](#) ([indico.modules.events.timetable.operations](#)), 256  
[modified\\_by\\_id](#) ([indico.modules.events.papers.models.comments.PaperReviewComment](#) attribute), 194  
[modified\\_dt](#) ([indico.modules.attachments.models.attachments.Attachment](#) attribute), 282  
[modified\\_dt](#) ([indico.modules.events.abstracts.models.abstracts.AbstractEvent](#) attribute), 146  
[modified\\_dt](#) ([indico.modules.events.abstracts.models.comments.AbstractComment](#) attribute), 149  
[modified\\_dt](#) ([indico.modules.events.abstracts.models.comments.AbstractComment](#) attribute), 149  
[modified\\_dt](#) ([indico.modules.events.abstracts.models.reviews.AbstractReview](#) attribute), 156  
[modified\\_dt](#) ([indico.modules.events.papers.models.comments.PaperReviewComment](#) attribute), 194  
[modified\\_dt](#) ([indico.modules.events.papers.models.reviews.PaperReviewComment](#) attribute), 200  
[modified\\_dt](#) ([indico.modules.auth.models.identities.Identity](#) attribute), 302

multipass\_group\_name (in- MultipleItemsField (class in in-  
 dico.modules.attachments.models.principals.AttachmentFolderPrincipal (class in in-  
 attribute), 286 MultiStringField (class in in-  
 multipass\_group\_name (in- dico.web.forms.fields), 342  
 dico.modules.attachments.models.principals.AttachmentPrincipal modules.users.models.users.UserTitle at-  
 attribute), 287 tribute), 276  
 multipass\_group\_name (in- N  
 dico.modules.categories.models.principals.CategoryPrincipal  
 attribute), 267 name (indico.core.oauth.models.applications.OAuthApplication  
 multipass\_group\_name (in- attribute), 307  
 dico.modules.events.contributions.models.principals.ContributionPrincipal  
 attribute), 176 name (indico.modules.categories.models.settings.CategorySetting  
 attribute), 268  
 multipass\_group\_name (in- name (indico.modules.designer.placeholders.CategoryTitlePlaceholder  
 dico.modules.events.models.principals.EventPrincipal attribute), 324  
 attribute), 131 name (indico.modules.designer.placeholders.EventDatesPlaceholder  
 multipass\_group\_name (in- attribute), 319  
 dico.modules.events.models.settings.EventSettingPrincipal (indico.modules.designer.placeholders.EventDescriptionPlaceholder  
 attribute), 136 attribute), 319  
 multipass\_group\_name (in- name (indico.modules.designer.placeholders.EventLogoPlaceholder  
 dico.modules.events.sessions.models.principals.SessionPrincipal attribute), 325  
 attribute), 244 name (indico.modules.designer.placeholders.EventOrgTextPlaceholder  
 multipass\_group\_name (in- attribute), 320  
 dico.modules.events.tracks.models.principals.TrackPrincipal (indico.modules.designer.placeholders.EventRoomPlaceholder  
 attribute), 261 attribute), 324  
 multipass\_group\_name (in- name (indico.modules.designer.placeholders.EventSpeakersPlaceholder  
 dico.modules.rb.models.blocking\_principals.BlockingPrincipal attribute), 324  
 attribute), 295 name (indico.modules.designer.placeholders.EventTitlePlaceholder  
 multipass\_group\_provider (in- attribute), 323  
 dico.modules.attachments.models.principals.AttachmentFolderPrincipal (indico.modules.designer.placeholders.EventVenuePlaceholder  
 attribute), 286 attribute), 324  
 multipass\_group\_provider (in- name (indico.modules.designer.placeholders.RegistrationAddressPlaceholder  
 dico.modules.attachments.models.principals.AttachmentPrincipal attribute), 323  
 attribute), 287 name (indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder  
 multipass\_group\_provider (in- attribute), 323  
 dico.modules.categories.models.principals.CategoryPrincipal (indico.modules.designer.placeholders.RegistrationAmountPlaceholder  
 attribute), 267 attribute), 322  
 multipass\_group\_provider (in- name (indico.modules.designer.placeholders.RegistrationCountryPlaceholder  
 dico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 321  
 attribute), 176 name (indico.modules.designer.placeholders.RegistrationEmailPlaceholder  
 multipass\_group\_provider (in- attribute), 322  
 dico.modules.events.models.principals.EventPrincipal name (indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder  
 attribute), 131 attribute), 321  
 multipass\_group\_provider (in- name (indico.modules.designer.placeholders.RegistrationFriendlyIDPlaceholder  
 dico.modules.events.models.settings.EventSettingPrincipal attribute), 322  
 attribute), 136 name (indico.modules.designer.placeholders.RegistrationFullNameNoTitle  
 multipass\_group\_provider (in- attribute), 320  
 dico.modules.events.sessions.models.principals.SessionPrincipal (indico.modules.designer.placeholders.RegistrationFullNameNoTitle  
 attribute), 244 attribute), 320  
 multipass\_group\_provider (in- name (indico.modules.designer.placeholders.RegistrationFullNameNoTitle  
 dico.modules.events.tracks.models.principals.TrackPrincipal attribute), 321  
 attribute), 261 name (indico.modules.designer.placeholders.RegistrationFullNameNoTitle  
 multipass\_group\_provider (in- attribute), 321  
 dico.modules.rb.models.blocking\_principals.BlockingPrincipal (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder  
 attribute), 295 attribute), 320

`name (indico.modules.designer.placeholders.RegistrationFullnamePlaceholder.attribute)`, 320

`name (indico.modules.designer.placeholders.RegistrationFullnamePlaceholder.events.abstracts.placeholders.TargetAbstractTitlePlaceholder.attribute)`, 163

`name (indico.modules.designer.placeholders.RegistrationFullnamePlaceholder.events.abstracts.placeholders.TargetSubmitterFirstNamePlaceholder.attribute)`, 320

`name (indico.modules.designer.placeholders.RegistrationFullnamePlaceholder.events.abstracts.placeholders.TargetSubmitterLastNamePlaceholder.attribute)`, 321

`name (indico.modules.designer.placeholders.RegistrationFullnamePlaceholder.events.abstracts.placeholders.TargetSubmitterNamePlaceholder.attribute)`, 321

`name (indico.modules.designer.placeholders.RegistrationPhonePlaceholder.attribute)`, 323

`name (indico.modules.designer.placeholders.RegistrationPositionPlaceholder.attribute)`, 323

`name (indico.modules.designer.placeholders.RegistrationPricePlaceholder.attribute)`, 322

`name (indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder.attribute)`, 322

`name (indico.modules.designer.placeholders.RegistrationTitlePlaceholder.attribute)`, 321

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.logs.renderers.EmailRenderer.attribute)`, 159

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.logs.renderers.EventLogRendererBase.attribute)`, 160

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.logs.renderers.SimpleRenderer.attribute)`, 161

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.models.references.ReferenceType.attribute)`, 159

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.models.settings.EventSetting.attribute)`, 160

`name (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.events.models.settings.EventSettingPrincipal.attribute)`, 160

`name (indico.modules.events.abstracts.placeholders.CoAuthorPlaceholder.events.papers.models.templates.PaperTemplate.attribute)`, 161

`name (indico.modules.events.abstracts.placeholders.ContributionTypePlaceholder.events.persons.placeholders.ContributionsPlaceholder.attribute)`, 164

`name (indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder.events.persons.placeholders.EmailPlaceholder.attribute)`, 164

`name (indico.modules.events.abstracts.placeholders.EventTitlePlaceholder.events.persons.placeholders.EventLinkPlaceholder.attribute)`, 159

`name (indico.modules.events.abstracts.placeholders.EventURLPlaceholder.events.persons.placeholders.EventTitlePlaceholder.attribute)`, 159

`name (indico.modules.events.abstracts.placeholders.JudgmentCorrelationPlaceholder.events.persons.placeholders.FirstNamePlaceholder.attribute)`, 164

`name (indico.modules.events.abstracts.placeholders.PrimaryAuthorPlaceholder.events.persons.placeholders.LastNamePlaceholder.attribute)`, 161

`name (indico.modules.events.abstracts.placeholders.SubmitterFirstNamePlaceholder.events.persons.placeholders.RegisterLinkPlaceholder.attribute)`, 162

`name (indico.modules.events.abstracts.placeholders.SubmitterLastNamePlaceholder.events.registration.models.forms.RegistrationForm.attribute)`, 162

`name (indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder.events.registration.placeholders.invitations.FirstNamePlaceholder.attribute)`, 161

`name (indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder.events.registration.placeholders.invitations.InvitationPlaceholder.attribute)`, 162

`name (indico.modules.events.abstracts.placeholders.TargetAbstractPlaceholder.events.registration.placeholders.invitations.LastNamePlaceholder.attribute)`, 162





not\_started (*indico.modules.events.abstracts.models.abstracts.AbstractRevision*), 247  
 attribute), 147  
 nth\_parent () (*indico.modules.categories.models.categories.Category* attribute), 266  
 note (*indico.modules.events.contributions.models.contributions.Contribution* attribute), 171  
 number (*indico.modules.events.papers.models.revisions.PaperRevision* attribute), 211  
 note (*indico.modules.events.contributions.models.subcontributions.SubContribution* attribute), 178  
 number (*indico.modules.rb.models.rooms.Room* attribute), 292  
 note (*indico.modules.events.models.events.Event* attribute), 126  
 note (*indico.modules.events.notes.models.notes.EventNoteRevision* attribute), 192  
 OAuth2AuthorizationCode (class in *indico.core.oauth.models.tokens*), 307  
 note (*indico.modules.events.sessions.models.blocks.SessionBlock* attribute), 242  
 OAuthApplication (class in *indico.core.oauth.models.applications*), 304  
 note (*indico.modules.events.sessions.models.sessions.Session* attribute), 241  
 OAuthApplicationUserLink (class in *indico.core.oauth.models.applications*), 307  
 note\_added (in module *indico.core.signals.event*), 85  
 note\_deleted (in module *indico.core.signals.event*), 85  
 OAuthToken (class in *indico.core.oauth.models.tokens*), 308  
 note\_id (*indico.modules.events.notes.models.notes.EventNoteRevision* attribute), 192  
 object (*indico.modules.events.models.persons.PersonLinkBase* attribute), 130  
 note\_id (*indico.modules.search.result\_schemas.EventNoteResultSchema* attribute), 73  
 object (*indico.modules.events.timetable.models.entries.TimetableEntry* attribute), 255  
 note\_modified (in module *indico.core.signals.event*), 85  
 object\_relationship\_name (in *indico.modules.events.abstracts.models.persons.AbstractPersonLink* attribute), 153  
 note\_restored (in module *indico.core.signals.event*), 85  
 object\_relationship\_name (in *indico.modules.events.contributions.models.persons.ContributionPersonLink* attribute), 175  
 notification\_before\_days (in *indico.modules.rb.models.rooms.Room* attribute), 291  
 object\_relationship\_name (in *indico.modules.events.contributions.models.persons.SubContributionPersonLink* attribute), 175  
 notification\_before\_days\_monthly (in *indico.modules.rb.models.rooms.Room* attribute), 291  
 object\_relationship\_name (in *indico.modules.events.models.persons.EventPersonLink* attribute), 129  
 notification\_before\_days\_weekly (in *indico.modules.rb.models.rooms.Room* attribute), 291  
 object\_relationship\_name (in *indico.modules.events.models.persons.PersonLinkBase* attribute), 130  
 notification\_emails (in *indico.modules.rb.models.rooms.Room* attribute), 291  
 object\_relationship\_name (in *indico.modules.events.sessions.models.persons.SessionBlockPersonLink* attribute), 243  
 notification\_sender\_address (in *indico.modules.events.registration.models.forms.RegistrationForm* attribute), 220  
 occurrences (*indico.modules.rb.models.reservations.Reservation* attribute), 298  
 notification\_sent (in *indico.modules.rb.models.reservation\_occurrences.ReservationOccurrence* attribute), 300  
 OccurrenceForm (class in *indico.web.forms.fields*), 340  
 notifications\_enabled (in *indico.modules.events.surveys.models.surveys.Survey* attribute), 246  
 old\_api\_keys (*indico.modules.users.models.users.User* attribute), 275  
 notifications\_enabled (in *indico.modules.rb.models.rooms.Room* attribute), 291  
 open () (*indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstracts* method), 148  
 open () (*indico.modules.events.papers.models.call\_for\_papers.CallForPapers* method), 193  
 notify\_managers (in *indico.modules.categories.models.categories.Category* attribute), 266  
 open () (*indico.modules.events.surveys.models.surveys.Survey* method), 247  
 notify\_participants (in *indico.modules.events.surveys.models.surveys.Survey* attribute), 157  
 open\_cfa () (in module *indico.modules.events.abstracts.operations*), 157

<b>Index</b>	<b>447</b>
--------------	------------

attribute), 242  
 own\_venue (indico.modules.events.sessions.models.sessions.Session attribute), 201  
 attribute), 241  
 own\_venue (indico.modules.events.timetable.models.breaks.Break attribute), 254  
 own\_venue\_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 171  
 own\_venue\_id (indico.modules.events.models.events.Event attribute), 126  
 own\_venue\_id (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 171  
 own\_venue\_id (indico.modules.events.sessions.models.sessions.Session attribute), 242  
 own\_venue\_id (indico.modules.events.sessions.models.sessions.Session attribute), 241  
 own\_venue\_id (indico.modules.events.timetable.models.breaks.Break attribute), 254  
 own\_venue\_name (in- attribute), 196  
 dico.modules.events.contributions.models.contributions.Contribution (class in in- attribute), 171  
 own\_venue\_name (in- attribute), 198  
 dico.modules.events.models.events.Event PaperCommentVisibility (class in in- attribute), 126  
 own\_venue\_name (in- attribute), 198  
 dico.modules.events.sessions.models.blocks.SessionBlock Competence (class in in- attribute), 242  
 own\_venue\_name (in- attribute), 195  
 dico.modules.events.sessions.models.sessions.Session PaperEmailSettingsField (class in in- attribute), 241  
 own\_venue\_name (in- PaperFile (class in in- attribute), 254  
 dico.modules.events.timetable.models.breaks.Break dico.modules.events.papers.models.files), 195  
 own\_visibility\_horizon (in- PaperJudgmentProxy (class in in- attribute), 266  
 dico.modules.categories.models.categories.Category dico.modules.events.papers.models.reviews), 199  
 owner (indico.modules.designer.models.templates.DesignerTemplate (class in in- attribute), 318  
 owner (indico.modules.rb.models.rooms.Room attribute), 292  
 owner\_id (indico.modules.rb.models.rooms.Room attribute), 292  
 PaperReview (class in in- attribute), 199  
 PaperReviewComment (class in in- attribute), 194  
 PaperReviewQuestion (class in in- attribute), 197  
 page (indico.modules.events.layout.models.menu.MenuEntry PaperReviewRating (class in in- attribute), 184  
 page (indico.modules.events.layout.models.menu.MenuEntryType dico.modules.events.papers.models.review\_ratings), 198  
 page\_id (indico.modules.events.layout.models.menu.MenuEntry PaperReviewType (class in in- attribute), 184  
 dico.modules.events.papers.models.reviews), 200  
 Paper (class in indico.modules.events.papers.models.papers), 196  
 PaperRevision (class in in- attribute), 200  
 paper (indico.modules.events.contributions.models.contributions.Contribution dico.modules.events.papers.models.revisions), 171  
 paper (indico.modules.events.papers.models.files.PaperFile PaperRevisionState (class in in- attribute), 196  
 dico.modules.events.papers.models.revisions), 202





person	(indico.modules.events.abstracts.models.persons.AbstractPersonLinkListField attribute), 153	person	(indico.modules.events.fields.EventPersonLinkListField attribute), 327
person	(indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 175	person	(indico.modules.events.fields.PersonLinkListFieldBase attribute), 175
person	(indico.modules.events.contributions.models.persons.SubContributionPersonLink attribute), 175	person	(indico.modules.events.fields.SessionBlockPersonLinkListField attribute), 175
person	(indico.modules.events.models.persons.EventPersonLink attribute), 129	person	(indico.modules.events.sessions.fields.SessionBlockPersonLinkListField attribute), 333
person	(indico.modules.events.models.persons.PersonLinkBase attribute), 130	person	(indico.modules.events.models.persons.PersonLinkDataMixin attribute), 130
person	(indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243	person	(indico.modules.events.models.persons.SessionBlockPersonLink attribute), 243
person_email	(indico.modules.events.agreements.models.agreements.Agreement attribute), 166	person_email	(indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 153
person_id	(indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 153	person_id	(indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 175
person_id	(indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 175	person_id	(indico.modules.events.contributions.models.persons.SubContributionPersonLink attribute), 175
person_id	(indico.modules.events.models.persons.EventPersonLink attribute), 129	person_id	(indico.modules.events.models.persons.EventPersonLink attribute), 129
person_id	(indico.modules.events.models.persons.PersonLinkBase attribute), 130	person_id	(indico.modules.events.models.persons.PersonLinkBase attribute), 130
person_id	(indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243	person_id	(indico.modules.events.models.persons.SessionBlockPersonLink attribute), 243
person_link_backref_name	(indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 153	person_link_backref_name	(indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243
person_link_backref_name	(indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 175	person_link_backref_name	(indico.modules.events.contributions.models.persons.SubContributionPersonLink attribute), 175
person_link_backref_name	(indico.modules.events.contributions.models.persons.SubContributionPersonLink attribute), 175	person_link_backref_name	(indico.modules.events.contributions.models.subcontributions.models.persons.SubContributionPersonLink attribute), 175
person_link_backref_name	(indico.modules.events.models.events.EventPersonLink attribute), 129	person_link_backref_name	(indico.modules.events.sessions.models.blocks.SessionBlockPersonLink attribute), 242
person_link_backref_name	(indico.modules.events.models.persons.PersonLinkBase attribute), 130	person_link_backref_name	(indico.modules.events.agreements.models.agreements.Agreement attribute), 166
person_link_backref_name	(indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243	person_link_backref_name	(indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243
person_link_cls	(indico.modules.events.abstracts.fields.AbstractPersonLinkListField attribute), 329	person_link_cls	(indico.modules.events.registration.models.form_fields.RegistrationFormItem attribute), 217
person_link_cls	(indico.modules.events.contributions.fields.ContributionPersonLinkListField attribute), 331	person_link_cls	(indico.modules.events.registration.models.form_fields.RegistrationFormItem attribute), 217
person_link_cls	(indico.modules.events.contributions.fields.SubContributionPersonLinkListField attribute), 331	person_link_cls	(indico.modules.events.registration.models.items.RegistrationFormItem attribute), 217
person_link_cls	(indico.modules.events.registration.models.items.RegistrationFormItem attribute), 217	person_link_cls	(indico.modules.events.registration.models.items.RegistrationFormItem attribute), 217

attribute), 225  
 personal\_data\_type (in-  
   dico.modules.events.registration.models.items.RegistrationFormBase  
   attribute), 226  
 personal\_data\_type (in-  
   dico.modules.events.registration.models.items.RegistrationFormBase  
   attribute), 226  
 PersonalDataType (class in in-  
   dico.modules.events.registration.models.items),  
   223  
 PersonLinkBase (class in in-  
   dico.modules.events.models.persons), 129  
 PersonLinkDataMixin (class in in-  
   dico.modules.events.models.persons), 130  
 PersonLinkListFieldBase (class in in-  
   dico.modules.events.fields), 328  
 PersonMixin (class in in-  
   dico.modules.users.models.users), 271  
 PersonNamePlaceholder (class in in-  
   dico.modules.events.agreements.placeholders),  
   168  
 persons (indico.modules.search.result\_schemas.ContributionResultSchema  
   attribute), 72  
 persons (indico.modules.search.result\_schemas.EventResultSchema  
   attribute), 71  
 PersonSchema (class in in-  
   dico.modules.search.result\_schemas), 73  
 phone (indico.modules.events.models.persons.EventPerson  
   attribute), 129  
 phone (indico.modules.events.models.persons.PersonLinkBase  
   attribute), 130  
 phone (indico.modules.events.registration.models.items.PersonalData  
   attribute), 223  
 phone (indico.modules.users.models.users.User  
   attribute), 275  
 Photo (class in indico.modules.rb.models.photos), 296  
 photo (indico.modules.rb.models.rooms.Room  
   attribute), 292  
 photo\_id (indico.modules.rb.models.rooms.Room  
   attribute), 292  
 picture (indico.modules.users.models.users.User  
   attribute), 275  
 picture\_metadata (in-  
   dico.modules.users.models.users.User  
   attribute), 275  
 picture\_source (in-  
   dico.modules.users.models.users.User  
   attribute), 275  
 plugin (indico.modules.events.layout.models.menu.MenuEntry  
   attribute), 184  
 plugin (indico.modules.events.layout.util.MenuEntryData  
   attribute), 186  
 plugin (indico.modules.events.logs.renderers.EventLogRenderBase  
   attribute), 189  
 plugin (indico.modules.events.payment.models.transactions.PaymentTran  
   attribute), 207  
 plugin (indico.modules.events.requests.base.RequestDefinitionBase  
   attribute), 239  
 plugin (indico.modules.vc.models.vc\_rooms.VCRoom  
   attribute), 312  
 plugin\_link (indico.modules.events.layout.models.menu.MenuEntry  
   attribute), 184  
 plugin\_url\_rule\_to\_js() (in module in-  
   dico.core.plugins), 80  
 PluginCategory (class in indico.core.plugins), 80  
 PLUGINS (built-in variable), 57  
 populate\_obj() (indico.web.forms.fields.JSONField  
   method), 335  
 position (indico.modules.categories.models.categories.Category  
   attribute), 266  
 position (indico.modules.events.abstracts.models.email\_templates.Abstr  
   attribute), 151  
 position (indico.modules.events.abstracts.models.review\_questions.Abstr  
   attribute), 154  
 position (indico.modules.events.contributions.models.fields.Contribution  
   attribute), 173  
 position (indico.modules.events.contributions.models.subcontributions.S  
   attribute), 178  
 position (indico.modules.events.layout.models.menu.MenuEntry  
   attribute), 184  
 position (indico.modules.events.papers.models.review\_questions.PaperR  
   attribute), 198  
 position (indico.modules.events.registration.models.form\_fields.Registra  
   attribute), 217  
 position (indico.modules.events.registration.models.form\_fields.Registra  
   attribute), 218  
 position (indico.modules.events.registration.models.items.PersonalData  
   attribute), 223  
 position (indico.modules.events.registration.models.items.RegistrationF  
   attribute), 224  
 position (indico.modules.events.registration.models.items.RegistrationF  
   attribute), 225  
 position (indico.modules.events.registration.models.items.RegistrationF  
   attribute), 226  
 position (indico.modules.events.registration.models.items.RegistrationF  
   attribute), 226  
 position (indico.modules.events.surveys.models.items.SurveyItem  
   attribute), 248  
 position (indico.modules.events.surveys.models.items.SurveyQuestion  
   attribute), 249  
 position (indico.modules.events.surveys.models.items.SurveySection  
   attribute), 250  
 position (indico.modules.events.surveys.models.items.SurveyText  
   attribute), 250  
 position (indico.modules.events.tracks.models.tracks.Track  
   attribute), 260  
 position (indico.modules.events.abstracts.models.abstracts.AbstractRev  
   attribute), 147

positive (indico.modules.events.logs.models.entries.EventLogKind attribute), 187  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.categories.models.categories.Category attribute), 266  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.abstracts.models.abstracts.Abstract attribute), 146  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.abstracts.models.reviews.AbstractReview attribute), 156  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 171  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.contributions.models.subcontributions.SubContribution attribute), 178  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.models.events.Event attribute), 126  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.papers.models.reviews.PaperReview attribute), 200  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.papers.models.revisions.PaperRevision attribute), 201  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.sessions.models.sessions.Session attribute), 241  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.surveys.models.items.SurveyItem attribute), 248  
 possible\_render\_modes (in- preferences (in module indico.core.signals.users), 90  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.timetable.models.breaks.Break attribute), 254  
 possible\_render\_modes (in- pre\_validate() (in- dico.modules.events.tracks.models.tracks.Track attribute), 260  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.abstracts.fields.AbstractField method), 328  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.abstracts.fields.AbstractPersonLinkField method), 329  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.abstracts.fields.EmailRuleListField method), 330  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.contributions.fields.ContributionPersonLinkField method), 331  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.fields.EventPersonLinkListField method), 327  
 pre\_validate() (in- pre\_validate() (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 169



[PRELOAD\\_EVENT\\_NOTES](#) (in- [principal\\_for](#) (in- [dico.modules.events.contributions.models.subcontributions.SubContributionCategories.models.principals.CategoryPrincipal](#) attribute), 177 [attribute](#)), 267  
[PRELOAD\\_EVENT\\_NOTES](#) (in- [principal\\_for](#) (in- [dico.modules.events.sessions.models.sessions.Session](#) [dico.modules.events.contributions.models.principals.Contribution](#) attribute), 240 [attribute](#)), 176  
[Previewer](#) (class in in- [principal\\_for](#) (in- [dico.modules.attachments.preview](#)), 288 [dico.modules.events.models.principals.EventPrincipal](#) attribute), 131  
[price](#) ([indico.modules.events.registration.models.registrations.Registration](#) [attribute](#)), 214 [principal\\_for](#) (in- [indico.modules.events.registration.models.registrations.RegistrationDates.events.sessions.models.principals.SessionPrincipal](#) attribute), 216 [attribute](#)), 244  
[price\\_adjustment](#) (in- [principal\\_for](#) (in- [dico.modules.events.registration.models.registrations.Registration](#) [dico.modules.events.tracks.models.principals.TrackPrincipal](#) attribute), 214 [attribute](#)), 261  
[primary](#) ([indico.modules.events.contributions.models.persons.Authors](#) [AuthorType](#) order (in- [attribute](#)), 174 [dico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 221  
[primary\\_authors](#) (in- [attribute](#)), 127 [dico.modules.events.models.persons.AuthorsSpeakersMixin](#) [principal\\_order](#) (in- [attribute](#)), 311 [dico.modules.groups.core.GroupProxy](#) at-  
[primary\\_email\\_changed](#) (in module in- [tribute](#)), 90 [principal\\_order](#) (in- [dico.core.signals.users](#)), 90 [principal\\_order](#) (in- [dico.modules.events.abstracts.placeholders](#)), 161 [dico.modules.networks.models.networks.IPNetworkGroup](#) attribute), 326  
[PrimaryAuthorsPlaceholder](#) (class in in- [tribute](#)), 161 [principal\\_order](#) (in- [dico.modules.events.models.persons.EventPerson](#) [dico.modules.users.models.users.User](#) at-  
[principal](#) ([attribute](#)), 129 [tribute](#)), 275  
[principal\\_backref\\_name](#) (in- [principal\\_type](#) (in- [dico.modules.attachments.models.principals.AttachmentFolderPrincipal](#) [dico.modules.events.registration.models.forms.RegistrationForm](#) attribute), 286 [attribute](#)), 221  
[principal\\_backref\\_name](#) (in- [principal\\_type](#) (in- [dico.modules.attachments.models.principals.AttachmentPrincipal](#) [dico.modules.networks.models.networks.IPNetworkGroup](#) attribute), 287 [attribute](#)), 326  
[principal\\_backref\\_name](#) (in- [principal\\_type](#) (in- [dico.modules.categories.models.principals.CategoryPrincipal](#) [dico.modules.users.models.users.User](#) attribute), 267 [tribute](#)), 275  
[principal\\_backref\\_name](#) (in- [PrincipalField](#) (class in [indico.web.forms.fields](#)), 344 [dico.modules.events.contributions.models.principals.ContributionPrincipal](#) attribute), 176 [PrincipalListField](#) (class in in- [dico.web.forms.fields](#)), 343  
[principal\\_backref\\_name](#) (in- [tribute](#)), 131 [principal\\_badge\\_template](#) (in module in- [dico.modules.events.models.principals.EventPrincipal](#) [dico.core.signals.event](#)), 86  
[principal\\_backref\\_name](#) (in- [private](#) ([indico.modules.events.surveys.models.surveys.Survey](#) [Principal](#) attribute), 247 [attribute](#)), 136 [process](#) (in module [indico.core.signals.rh](#)), 90  
[principal\\_backref\\_name](#) (in- [process\\_args](#) (in module [indico.core.signals.rh](#)), 90 [dico.modules.events.sessions.models.principals.SessionPrincipal](#) attribute), 244 [data\(\)](#) (in- [dico.modules.categories.fields.CategoryField](#) method), 333  
[principal\\_backref\\_name](#) (in- [tribute](#)), 261 [Principal\\_data\(\)](#) (in- [dico.modules.events.tracks.models.principals.TrackPrincipal](#) [dico.modules.networks.fields.MultiIPNetworkField](#) method), 333  
[principal\\_backref\\_name](#) (in- [tribute](#)), 295 [Principal\\_data\(\)](#) (in- [dico.modules.rb.models.blocking\\_principals.BlockingPrincipal](#) [dico.web.forms.fields.IndicoEmailRecipientsField](#) attribute), 295

<i>method</i> ), 348		<i>method</i> ), 346	
<code>process_data()</code> ( <i>in-dico.web.forms.fields.IndicoPalettePickerField method</i> ), 339		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoSinglePalettePickerField method</i> ), 339	
<code>process_data()</code> ( <i>in-dico.web.forms.fields.IndicoStaticTextField method</i> ), 338		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoWeekDayRepetitionField method</i> ), 347	
<code>process_data()</code> ( <i>in-dico.web.forms.fields.IndicoTimezoneSelectField method</i> ), 341		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.JSONField method</i> ), 335	
<code>process_form_data()</code> ( <i>in-dico.modules.users.ext.ExtraUserPreferences method</i> ), 281		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.MultipleItemsField method</i> ), 343	
<code>process_formdata()</code> ( <i>in-dico.modules.categories.fields.CategoryField method</i> ), 333		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.MultiStringField method</i> ), 342	
<code>process_formdata()</code> ( <i>in-dico.modules.events.fields.EventPersonListField method</i> ), 327		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.OccurrencesField method</i> ), 341	
<code>process_formdata()</code> ( <i>in-dico.modules.events.fields.ReferencesField method</i> ), 328		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.OverrideMultipleItemsField method</i> ), 343	
<code>process_formdata()</code> ( <i>in-dico.modules.events.papers.fields.PaperEmailSettingsField method</i> ), 333		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.PrincipalField method</i> ), 344	
<code>process_formdata()</code> ( <i>in-dico.modules.networks.fields.MultiIPNetworkField method</i> ), 334		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.PrincipalListField method</i> ), 344	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.EditableFileField method</i> ), 345		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.RelativeDeltaField method</i> ), 346	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.EmailListField method</i> ), 338		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.TextListField method</i> ), 337	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.FileField method</i> ), 342		<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.TimeDeltaField method</i> ), 340	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.HiddenEnumField method</i> ), 341		<code>processed_by_id</code> ( <i>in-dico.modules.events.requests.models.requests.Request attribute</i> ), 237	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.HiddenFieldList method</i> ), 336		<code>processed_by_user</code> ( <i>in-dico.modules.events.requests.models.requests.Request attribute</i> ), 237	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoDateTimeField method</i> ), 340		<code>processed_dt</code> ( <i>in-dico.modules.events.requests.models.requests.Request attribute</i> ), 237	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoLocationField method</i> ), 345		<code>prof</code> ( <i>in-dico.modules.users.models.users.UserTitle attribute</i> ), 276	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoPalettePickerField method</i> ), 339		<code>PROFILE</code> (built-in variable), 52	
<code>process_formdata()</code> ( <i>in-dico.web.forms.fields.IndicoSelectMultipleCheckboxBooleanField method</i> ), 341		<code>ProfilePictureSource</code> (class <i>in-dico.modules.users.models.users</i> ), 272	
		<code>proposal</code> ( <i>in-dico.modules.events.models.reviews.ProposalRevisionMixin attribute</i> ), 134	
		<code>proposal_attr</code> ( <i>in-dico.modules.events.models.reviews.ProposalRevisionMixin attribute</i> ), 134	

attribute), 135  
 proposal\_attr (in- protection\_mode (in-  
   dico.modules.events.papers.models.revisions.PaperRevisiondico.modules.events.contributions.models.contributions.Contributio  
   attribute), 201 attribute), 172  
 proposal\_type (in- protection\_mode (in-  
   dico.modules.events.abstracts.models.abstracts.Abstract dico.modules.events.models.events.Event  
   attribute), 146 attribute), 126  
 proposal\_type (in- protection\_mode (in-  
   dico.modules.events.models.reviews.ProposalMixin dico.modules.events.sessions.models.sessions.Session  
   attribute), 134 attribute), 241  
 proposal\_type (in- protection\_mode (in-  
   dico.modules.events.papers.models.papers.Paper dico.modules.events.tracks.models.tracks.Track  
   attribute), 197 attribute), 260  
 ProposalCommentMixin (class in in- protection\_mode (in-  
   dico.modules.events.models.reviews), 133 dico.modules.rb.models.rooms.Room attribute),  
 ProposalGroupProxy (class in in- 292  
   dico.modules.events.models.reviews), 133 protection\_parent (in-  
 ProposalMixin (class in in- dico.modules.attachments.models.attachments.Attachment  
   dico.modules.events.models.reviews), 133 attribute), 282  
 ProposalReviewMixin (class in in- protection\_parent (in-  
   dico.modules.events.models.reviews), 134 dico.modules.attachments.models.folders.AttachmentFolder  
 ProposalRevisionMixin (class in in- attribute), 285  
   dico.modules.events.models.reviews), 134 protection\_parent (in-  
 proposed\_action (in- dico.modules.categories.models.categories.Category  
   dico.modules.events.abstracts.models.reviews.AbstractReviewattribute), 266  
   attribute), 156 protection\_parent (in-  
 proposed\_action (in- dico.modules.events.contributions.models.contributions.Contributio  
   dico.modules.events.papers.models.reviews.PaperReview attribute), 172  
   attribute), 200 protection\_parent (in-  
 proposed\_contribution\_type (in- dico.modules.events.models.events.Event  
   dico.modules.events.abstracts.models.reviews.AbstractReviewattribute), 126  
   attribute), 156 protection\_parent (in-  
 proposed\_contribution\_type\_id (in- dico.modules.events.sessions.models.sessions.Session  
   dico.modules.events.abstracts.models.reviews.AbstractReviewattribute), 241  
   attribute), 156 protection\_parent (in-  
 proposed\_related\_abstract (in- dico.modules.rb.models.rooms.Room attribute),  
   dico.modules.events.abstracts.models.reviews.AbstractReview292  
   attribute), 156 provider (indico.modules.auth.models.identities.Identity  
 proposed\_related\_abstract\_id (in- attribute), 302  
   dico.modules.events.abstracts.models.reviews.AbstpractReview (indico.modules.events.payment.models.transactions.PaymentT  
   attribute), 156 attribute), 207  
 proposed\_tracks (in- PROVIDER\_MAP (built-in variable), 48  
   dico.modules.events.abstracts.models.reviews.AbstpractReviewattr (indico.modules.events.papers.models.papers.Paper  
   attribute), 156 attribute), 197  
 protection\_changed (in module in- proxy (indico.modules.groups.models.groups.LocalGroup  
   dico.core.signals.acl), 81 attribute), 310  
 protection\_mode (in- proxy\_to\_reservation\_if\_last\_valid\_occurrence()  
   dico.modules.attachments.models.attachments.Attachment (in module indico.modules.rb.models.util), 301  
   attribute), 282 public (indico.modules.events.contributions.models.fields.ContributionFi  
 protection\_mode (in- attribute), 174  
   dico.modules.attachments.models.folders.AttachmentFolderregform\_access (in-  
   attribute), 285 dico.modules.events.models.events.Event  
 protection\_mode (in- attribute), 126  
   dico.modules.categories.models.categories.Categorypublic\_state (indico.modules.events.abstracts.models.abstracts.Abstpr

[attribute](#)), 146  
[PUBLIC\\_SUPPORT\\_EMAIL](#) (*built-in variable*), 53  
[publish\\_checkin\\_enabled](#) (*in-*  
[dico.modules.events.registration.models.forms.RegistrationForm](#)  
[attribute](#)), 221  
[publish\\_registration\\_count](#) (*in-*  
[dico.modules.events.registration.models.forms.RegistrationForm](#)  
[attribute](#)), 221  
[publish\\_registrations\\_enabled](#) (*in-*  
[dico.modules.events.registration.models.forms.RegistrationForm](#)  
[attribute](#)), 221  
[published\\_registrations](#) (*in-*  
[dico.modules.events.models.events.Event](#)  
[attribute](#)), 126

## Q

[query](#) (*indico.modules.categories.settings.CategorySettingsProxy*  
[attribute](#)), 270  
[query](#) (*indico.modules.events.settings.EventSettingsProxy*  
[attribute](#)), 143, 206  
[query](#) (*indico.modules.users.models.settings.UserSettingsProxy*  
[attribute](#)), 278  
[query\\_active\\_surveys\(\)](#) (*in module in-*  
[dico.modules.events.surveys.util](#)), 252  
[query\\_personal\\_tokens\(\)](#) (*in-*  
[dico.modules.users.models.users.User](#)  
[method](#)), 275  
[question](#) (*indico.modules.events.abstracts.models.review\_ratings.AbstractReviewRating*  
[attribute](#)), 154  
[question](#) (*indico.modules.events.papers.models.review\_ratings.PaperReviewRating*  
[attribute](#)), 198  
[question](#) (*indico.modules.events.surveys.models.items.SurveyItemType*  
[attribute](#)), 248  
[question](#) (*indico.modules.events.surveys.models.submissions.SurveyAnswer*  
[attribute](#)), 251  
[question\\_class](#) (*in-*  
[dico.modules.events.abstracts.models.review\\_ratings.AbstractReviewRating](#)  
[attribute](#)), 154  
[question\\_class](#) (*in-*  
[dico.modules.events.papers.models.review\\_ratings.PaperReviewRating](#)  
[attribute](#)), 198  
[question\\_id](#) (*indico.modules.events.abstracts.models.review\_ratings.AbstractReviewRating*  
[attribute](#)), 154  
[question\\_id](#) (*indico.modules.events.papers.models.review\_ratings.PaperReviewRating*  
[attribute](#)), 198  
[question\\_id](#) (*indico.modules.events.surveys.models.submissions.SurveyAnswer*  
[attribute](#)), 251  
[questions](#) (*indico.modules.events.surveys.models.surveys.Survey*  
[attribute](#)), 247

## R

[radio\\_widget](#) (*indico.web.forms.fields.IndicoProtectionField*  
[attribute](#)), 345

[rating\\_range](#) (*indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstracts*  
[attribute](#)), 148  
[rating\\_range](#) (*indico.modules.events.papers.models.call\_for\_papers.CallForPapers*  
[attribute](#)), 193  
[RatingReviewField](#) (*class in in-*  
[dico.modules.events.fields](#)), 328  
[register\\_user\\_access\(\)](#) (*in module in-*  
[dico.modules.rb.util](#)), 301  
[rb\\_is\\_admin\(\)](#) (*in module indico.modules.rb.util*), 301  
[read\\_access](#) (*indico.modules.categories.models.principals.CategoryPrincipal*  
[attribute](#)), 267  
[read\\_access](#) (*indico.modules.events.contributions.models.principals.ContributionPrincipal*  
[attribute](#)), 176  
[read\\_access](#) (*indico.modules.events.models.principals.EventPrincipal*  
[attribute](#)), 131  
[read\\_access](#) (*indico.modules.events.sessions.models.principals.SessionPrincipal*  
[attribute](#)), 244  
[read\\_access](#) (*indico.modules.events.tracks.models.principals.TrackPrincipal*  
[attribute](#)), 261  
[ready\\_to\\_open](#) (*in-*  
[dico.modules.events.surveys.models.surveys.SurveyState](#)  
[attribute](#)), 247  
[real\\_visibility\\_horizon](#) (*in-*  
[dico.modules.categories.models.categories.Category](#)  
[attribute](#)), 266  
[realm](#) (*indico.modules.events.logs.models.entries.EventLogEntry*  
[attribute](#)), 187  
[reason](#) (*indico.modules.events.agreements.models.agreements.Agreement*  
[attribute](#)), 166  
[reason](#) (*indico.modules.rb.models.blockings.Blocking*  
[attribute](#)), 294  
[recipients](#) (*indico.modules.events.abstracts.models.email\_logs.AbstractEmailLog*  
[attribute](#)), 150  
[recipients](#) (*indico.modules.events.reminders.models.reminders.EventReminder*  
[attribute](#)), 235  
[redirect\\_to\\_login\(\)](#) (*in module in-*  
[dico.modules.auth.util](#)), 304  
[redirect\\_uri](#) (*indico.core.oauth.models.tokens.OAuth2AuthorizationCode*  
[attribute](#)), 308  
[redirect\\_uris](#) (*in-*  
[dico.core.oauth.models.applications.OAuthApplication](#)  
[attribute](#)), 307  
[REDIS\\_CACHE\\_URL](#) (*built-in variable*), 49  
[reference\\_backref\\_name](#) (*in-*  
[indico.modules.events.contributions.models.references.ContributionReference](#)  
[attribute](#)), 177  
[reference\\_backref\\_name](#) (*in-*  
[dico.modules.events.contributions.models.references.SubContributionReference](#)  
[attribute](#)), 177  
[reference\\_backref\\_name](#) (*in-*  
[dico.modules.events.models.references.EventReference](#)  
[attribute](#)), 131  
[reference\\_backref\\_name](#) (*in-*

[dico.modules.events.models.references.ReferenceModelBase](#)[dico.modules.events.util](#)), 141  
[attribute](#)), 132      [register\\_transaction\(\)](#) (in module [in-](#)  
[reference\\_type](#) (in- [dico.modules.events.payment.util](#)), 208  
[dico.modules.events.contributions.models.references.ContributionReference](#) (in module [in-](#)  
[attribute](#)), 177      [dico.modules.auth.util](#)), 304  
[reference\\_type](#) (in- [registered](#) (in module [indico.core.signals.users](#)), 90  
[dico.modules.events.contributions.models.references.SubContributionReference](#) (in-  
[attribute](#)), 177      [dico.modules.events.layout.models.menu.MenuEntry](#)  
[reference\\_type](#) (in- [attribute](#)), 184  
[dico.modules.events.models.references.EventReference](#)[registerLinkPlaceholder](#) (class in [in-](#)  
[attribute](#)), 131      [dico.modules.events.persons.placeholders](#)),  
[reference\\_type](#) (in- 212  
[dico.modules.events.models.references.ReferenceModelBase](#) (class in [in-](#)  
[attribute](#)), 132      [dico.modules.events.registration.models.registrations](#)),  
[reference\\_type\\_id](#) (in- 212  
[dico.modules.events.contributions.models.references.ContributionReference](#)[indico.modules.events.payment.models.transactions.Pay](#)  
[attribute](#)), 177      [attribute](#)), 207  
[reference\\_type\\_id](#) (in- [registration](#) ([indico.modules.events.registration.models.invitations.Re](#)  
[dico.modules.events.contributions.models.references.SubContributionReference](#)  
[attribute](#)), 177      [registration\\_checkin\\_updated](#) (in module [in-](#)  
[reference\\_type\\_id](#) (in- [dico.core.signals.event](#)), 86  
[dico.modules.events.models.references.EventReference](#)[registration\\_created](#) (in module [in-](#)  
[attribute](#)), 131      [dico.core.signals.event](#)), 86  
[reference\\_type\\_id](#) (in- [registration\\_deleted](#) (in module [in-](#)  
[dico.modules.events.models.references.ReferenceModelBase](#)[dico.core.signals.event](#)), 86  
[attribute](#)), 132      [registration\\_form](#) (in-  
[ReferenceModelBase](#) (class in [in-](#) [dico.modules.attachments.models.principals.AttachmentFolderPr](#)  
[dico.modules.events.models.references](#)), [attribute](#)), 286  
131      [registration\\_form](#) (in-  
[references](#) ([indico.modules.events.contributions.models.contributions.attachments.models.principals.AttachmentPrincipal](#)  
[attribute](#)), 172      [attribute](#)), 287  
[references](#) ([indico.modules.events.contributions.models.subcontributions.SubContribution](#) (in-  
[attribute](#)), 178      [dico.modules.categories.models.principals.CategoryPrincipal](#)  
[references](#) ([indico.modules.events.models.events.Event](#) [attribute](#)), 267  
[attribute](#)), 126      [registration\\_form](#) (in-  
[ReferencesField](#) (class in [in-](#) [dico.modules.events.contributions.models.principals.Contribution](#)  
[dico.modules.events.fields](#)), 328      [attribute](#)), 176  
[ReferenceType](#) (class in [in-](#) [registration\\_form](#) (in-  
[dico.modules.events.models.references](#)), [dico.modules.events.models.principals.EventPrincipal](#)  
132      [attribute](#)), 131  
[refresh\\_event\\_persons\(\)](#) (in- [registration\\_form](#) (in-  
[dico.modules.events.models.events.Event](#) [dico.modules.events.models.settings.EventSettingPrincipal](#)  
[method](#)), 126      [attribute](#)), 136  
[register\\_event\\_time\\_change\(\)](#) (in module [in-](#) [registration\\_form](#) (in-  
[dico.modules.events.util](#)), 140      [dico.modules.events.sessions.models.principals.SessionPrincipal](#)  
[register\\_link\\_events\(\)](#) (in- [attribute](#)), 244  
[dico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#)[registration\\_form](#) (in-  
[class method](#)), 313      [dico.modules.events.tracks.models.principals.TrackPrincipal](#)  
[register\\_location\\_change\(\)](#) (in module [in-](#) [attribute](#)), 261  
[dico.modules.events.util](#)), 140      [registration\\_form](#) (in-  
[register\\_login\(\)](#) (in- [dico.modules.rb.models.blocking\\_principals.BlockingPrincipal](#)  
[dico.modules.auth.models.identities.Identity](#) [attribute](#)), 295  
[method](#)), 303      [registration\\_form\\_created](#) (in module [in-](#)  
[register\\_time\\_change\(\)](#) (in module [in-](#) [dico.core.signals.event](#)), 86



registration\_form\_deleted (in module `indico.core.signals.event`), 86  
 registration\_form\_created (in module `indico.core.signals.event`), 86  
 registration\_form\_id (in `indico.modules.attachments.models.principals.AttachmentFolderPrincipal` attribute), 286  
 registration\_form\_id (in `indico.modules.events.registration.models.invitations.RegistrationInvitationPrincipal` attribute), 222  
 registration\_form\_id (in `indico.modules.events.registration.models.registrations.RegistrationFormSectionPrincipal` attribute), 216  
 registration\_form\_id (in `indico.modules.events.registration.models.forms.RegistrationFormSectionPrincipal` attribute), 217  
 registration\_form\_id (in `indico.modules.events.models.principals.EventPrincipal` attribute), 131  
 registration\_form\_id (in `indico.modules.events.models.settings.EventSettingPrincipal` attribute), 136  
 registration\_form\_id (in `indico.modules.events.registration.models.form_fields.RegistrationFormFieldPrincipal` attribute), 217  
 registration\_form\_id (in `indico.modules.events.registration.models.form_fields.RegistrationFormFieldPrincipal` attribute), 219  
 registration\_form\_id (in `indico.modules.events.registration.models.invitations.RegistrationInvitationPrincipal` attribute), 222  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 224  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 225  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 226  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 219  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 226  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 217  
 registration\_form\_id (in `indico.modules.events.registration.models.items.RegistrationFormSectionPrincipal` attribute), 218  
 registration\_form\_id (in `indico.modules.events.sessions.models.principals.SessionPrincipal` attribute), 244  
 registration\_form\_id (in `indico.modules.events.tracks.models.principals.TrackPrincipal` attribute), 261  
 registration\_form\_id (in `indico.modules.rb.models.blocking_principals.BlockingPrincipal` attribute), 224  
 registration\_form\_wtform\_created (in module `indico.core.signals.event`), 86  
 registration\_id (in `indico.modules.events.payment.models.transactions.PaymentTransactionPrincipal` attribute), 207  
 registration\_id (in `indico.modules.events.registration.models.invitations.RegistrationInvitationPrincipal` attribute), 222  
 registration\_id (in `indico.modules.events.registration.models.registrations.RegistrationFormSectionPrincipal` attribute), 216  
 registration\_limit (in `indico.modules.events.registration.models.forms.RegistrationFormSectionPrincipal` attribute), 217  
 registration\_personal\_data\_modified (in module `indico.core.signals.event`), 86  
 registration\_requested (in module `indico.core.signals.users`), 90  
 registration\_state\_updated (in module `indico.core.signals.event`), 86  
 registration\_updated (in module `indico.core.signals.event`), 86  
 RegistrationAffiliationPlaceholder (class in `indico.modules.designer.placeholders`), 323  
 RegistrationAmountPlaceholder (class in `indico.modules.designer.placeholders`), 322  
 RegistrationCountryPlaceholder (class in `indico.modules.designer.placeholders`), 323  
 RegistrationData (class in `indico.modules.events.registration.models.registrations`), 215  
 RegistrationEmailPlaceholder (class in `indico.modules.designer.placeholders`), 322  
 RegistrationFirstNamePlaceholder (class in `indico.modules.designer.placeholders`), 321  
 RegistrationFormSection (class in `indico.modules.events.registration.models.forms`), 219  
 RegistrationFormTextFormField (class in `indico.modules.events.registration.models.form_fields`), 217  
 RegistrationFormFieldData (class in `indico.modules.events.registration.models.form_fields`), 218  
 RegistrationFormItem (class in `indico.modules.events.registration.models.items`), 223  
 RegistrationFormItemType (class in `indico.modules.events.registration.models.items`), 224  
 RegistrationFormPersonalDataField (class in `indico.modules.events.registration.models.form_fields`), 217

- (class in in- 232  
*indico.modules.events.registration.models.form\_fields*), 218
- RegistrationFormPersonalDataSection (class in in-  
*indico.modules.events.registration.models.items*), 224
- RegistrationFormSection (class in in-  
*indico.modules.events.registration.models.items*), 225
- RegistrationFormText (class in in-  
*indico.modules.events.registration.models.items*), 226
- RegistrationFriendlyIDPlaceholder (class  
in *indico.modules.designer.placeholders*), 322
- RegistrationFullNameNoTitlePlaceholder (class  
in in-  
*indico.modules.designer.placeholders*), 320
- RegistrationFullNameNoTitlePlaceholderB (class  
in in-  
*indico.modules.designer.placeholders*), 320
- RegistrationFullNameNoTitlePlaceholderC (class  
in in-  
*indico.modules.designer.placeholders*), 321
- RegistrationFullNameNoTitlePlaceholderD (class  
in in-  
*indico.modules.designer.placeholders*), 321
- RegistrationFullNamePlaceholder (class in  
*indico.modules.designer.placeholders*), 319
- RegistrationFullNamePlaceholderB (class in  
*indico.modules.designer.placeholders*), 320
- RegistrationFullNamePlaceholderC (class in  
*indico.modules.designer.placeholders*), 320
- RegistrationFullNamePlaceholderD (class in  
*indico.modules.designer.placeholders*), 321
- RegistrationInvitation (class in in-  
*indico.modules.events.registration.models.invitations*), 221
- RegistrationLastNamePlaceholder (class in  
*indico.modules.designer.placeholders*), 321
- RegistrationPhonePlaceholder (class in in-  
*indico.modules.designer.placeholders*), 323
- RegistrationPositionPlaceholder (class in  
*indico.modules.designer.placeholders*), 323
- RegistrationPricePlaceholder (class in in-  
*indico.modules.designer.placeholders*), 322
- RegistrationRequest (class in in-  
*indico.modules.auth.models.registration\_requests*), 303
- registrations (in-  
*indico.modules.events.registration.models.forms.RegistrationForm* attribute), 221
- RegistrationSettingsProxy (class in in-  
*indico.modules.events.registration.settings*),
- RegistrationState (class in in-  
*indico.modules.events.registration.models.registrations*), 216
- RegistrationTicketQRPlaceholder (class in  
*indico.modules.designer.placeholders*), 321
- RegistrationTitlePlaceholder (class in in-  
*indico.modules.designer.placeholders*), 321
- reject (*indico.modules.events.abstracts.models.reviews.AbstractAction*  
attribute), 154
- reject (*indico.modules.events.papers.models.reviews.PaperAction*  
attribute), 198
- reject (*indico.modules.events.payment.models.transactions.TransactionA*  
attribute), 208
- reject () (*indico.modules.events.agreements.models.agreements.Agreem*  
method), 167
- reject () (*indico.modules.events.requests.base.RequestDefinitionBase*  
class method), 239
- reject () (*indico.modules.rb.models.blocked\_rooms.BlockedRoom*  
method), 294
- reject () (*indico.modules.rb.models.reservation\_occurrences.Reservatio*  
method), 300
- reject () (*indico.modules.rb.models.reservations.Reservation*  
method), 298
- rejected (*indico.modules.events.abstracts.models.abstracts.AbstractPub*  
attribute), 147
- rejected (*indico.modules.events.abstracts.models.abstracts.AbstractStat*  
attribute), 148
- rejected (*indico.modules.events.agreements.models.agreements.Agreem*  
attribute), 167
- rejected (*indico.modules.events.agreements.models.agreements.Agreem*  
attribute), 167
- rejected (*indico.modules.events.papers.models.revisions.PaperRevision*  
attribute), 202
- rejected (*indico.modules.events.payment.models.transactions.Transaction*  
attribute), 208
- rejected (*indico.modules.events.registration.models.registrations.Registr*  
attribute), 217
- rejected (*indico.modules.events.requests.models.requests.RequestState*  
attribute), 237
- rejected (*indico.modules.rb.models.blocked\_rooms.BlockedRoomState*  
attribute), 294
- rejected (*indico.modules.rb.models.reservation\_occurrences.Reservatio*  
attribute), 301
- rejected (*indico.modules.rb.models.reservations.ReservationState*  
attribute), 299
- rejected\_by (*indico.modules.rb.models.blocked\_rooms.BlockedRoom*  
attribute), 294
- rejected\_on\_behalf (in-  
*indico.modules.events.agreements.models.agreements.AgreementSta*  
attribute), 167
- rejection\_reason (in-  
*indico.modules.events.registration.models.registrations.Registration*  
attribute), 214

`rejection_reason` (in- `class method`), 160  
`dico.modules.rb.models.blocked_rooms.BlockedRoom` `render ()` (`indico.modules.events.abstracts.placeholders.CoAuthorsPlaceholder` `class method`), 161  
`attribute`), 294  
`rejection_reason` (in- `render ()` (`indico.modules.events.abstracts.placeholders.ContributionTypePlaceholder` `class method`), 164  
`dico.modules.rb.models.reservation_occurrences.ReservationOccurrence` `render ()` (`indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder` `class method`), 164  
`attribute`), 300  
`rejection_reason` (in- `class method`), 164  
`dico.modules.rb.models.reservations.Reservation` `render ()` (`indico.modules.events.abstracts.placeholders.EventTitlePlaceholder` `class method`), 159  
`attribute`), 298  
`RejectionReasonPlaceholder` (class in in- `render ()` (`indico.modules.events.abstracts.placeholders.EventURLPlaceholder` `class method`), 159  
`dico.modules.events.registration.placeholders.registrations`) `render ()` (`indico.modules.events.abstracts.placeholders.JudgmentCommentPlaceholder` `class method`), 164  
`231`  
`RelativeDeltaField` (class in in- `class method`), 164  
`dico.web.forms.fields`), 346  
`remove_principal ()` (in- `class method`), 161  
`dico.modules.events.settings.EventACLProxy` `render ()` (`indico.modules.events.abstracts.placeholders.SubmitterFirstNamePlaceholder` `class method`), 162  
`method`), 142, 205  
`remove_room_spritesheet_photo ()` (in mod- `render ()` (`indico.modules.events.abstracts.placeholders.SubmitterLastNamePlaceholder` `class method`), 162  
`ule indico.modules.rb.util`), 301  
`render ()` (`indico.modules.designer.placeholders.CategoryTitlePlaceholder` `indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder` `class method`), 324  
`class method`), 161  
`render ()` (`indico.modules.designer.placeholders.EventDatesPlaceholder` `indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder` `class method`), 319  
`class method`), 162  
`render ()` (`indico.modules.designer.placeholders.EventDescriptionPlaceholder` `indico.modules.events.abstracts.placeholders.TargetAbstractIDPlaceholder` `class method`), 319  
`class method`), 162  
`render ()` (`indico.modules.designer.placeholders.EventLogoPlaceholder` `indico.modules.events.abstracts.placeholders.TargetAbstractTitlePlaceholder` `class method`), 325  
`class method`), 163  
`render ()` (`indico.modules.designer.placeholders.EventOrgTextPlaceholder` `indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder` `class method`), 320  
`class method`), 163  
`render ()` (`indico.modules.designer.placeholders.EventRoomPlaceholder` `indico.modules.events.abstracts.placeholders.TargetSubmitterTitlePlaceholder` `class method`), 324  
`class method`), 163  
`render ()` (`indico.modules.designer.placeholders.EventSpeakerPlaceholder` `indico.modules.events.abstracts.placeholders.TargetSubmitterURLPlaceholder` `class method`), 324  
`class method`), 163  
`render ()` (`indico.modules.designer.placeholders.EventTitlePlaceholder` `indico.modules.events.agreements.models.agreements.AgreementPlaceholder` `class method`), 323  
`class method`), 167  
`render ()` (`indico.modules.designer.placeholders.EventVenuePlaceholder` `indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder` `class method`), 324  
`class method`), 168  
`render ()` (`indico.modules.designer.placeholders.RegistrationAmountPlaceholder` `indico.modules.events.agreements.placeholders.PersonNamePlaceholder` `class method`), 322  
`class method`), 168  
`render ()` (`indico.modules.designer.placeholders.RegistrationPricePlaceholder` `indico.modules.events.logs.models.entries.EventLogEntry` `class method`), 322  
`class method`), 187  
`render ()` (`indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder` `indico.modules.events.persons.placeholders.ContributionsPlaceholder` `class method`), 322  
`class method`), 210  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder` `indico.modules.events.persons.placeholders.EmailPlaceholder` `class method`), 159  
`class method`), 210  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder` `indico.modules.events.persons.placeholders.EventLinkPlaceholder` `class method`), 160  
`class method`), 211  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractSessionPlaceholder` `indico.modules.events.persons.placeholders.EventTitlePlaceholder` `class method`), 161  
`class method`), 211  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder` `indico.modules.events.persons.placeholders.FirstNamePlaceholder` `class method`), 160  
`class method`), 211  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractTrackPlaceholder` `indico.modules.events.persons.placeholders.LastNamePlaceholder` `class method`), 160  
`class method`), 211  
`render ()` (`indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder` `indico.modules.events.persons.placeholders.RegisterLinkPlaceholder` `class method`), 160  
`class method`), 211



class method), 212  
 render () (indico.modules.events.registration.placeholders.invitations.models.invitations.InvitationPlaceholder  
 class method), 231  
 render () (indico.modules.events.registration.placeholders.invitations.models.invitations.InvitationPlaceholder  
 class method), 232  
 render () (indico.modules.events.registration.placeholders.invitations.models.invitations.InvitationPlaceholder  
 class method), 232  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 229  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 229  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 230  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 230  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 230  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 231  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 231  
 render () (indico.modules.events.registration.placeholders.registrations.models.registrations.RegistrationPlaceholder  
 class method), 231  
 render\_archive () (in module indico.modules.events.contributions.util), 181  
 render\_base\_price () (in module indico.modules.events.registration.models.forms.RegistrationForm), 221  
 render\_base\_price () (in module indico.modules.events.registration.models.registrations.RegistrationForm), 214  
 render\_buttons () (in module indico.modules.vc.plugins.VCPluginMixin), 315  
 render\_changes () (in module indico.modules.events.logs.util), 188  
 render\_details () (in module indico.modules.events.payment.models.transactions.PaymentTransaction), 207  
 render\_entry () (in module indico.modules.events.logs.renderers.EventLogRendererBase), 189  
 render\_entry\_info\_balloon () (in module indico.modules.events.timetable.util), 258  
 render\_event\_buttons () (in module indico.modules.vc.plugins.VCPluginMixin), 315  
 render\_form () (in module indico.modules.events.requests.base.RequestDefinitionBase), 239  
 render\_form () (in module indico.modules.vc.plugins.VCPluginMixin), 315  
 render\_info\_box () (in module indico.modules.vc.plugins.VCPluginMixin), 315  
 render\_manage\_event\_info\_box () (in module indico.modules.vc.plugins.VCPluginMixin), 316  
 render\_name\_placeholder () (in module indico.modules.events.contributions.models.contributions.Category), 266  
 render\_registration\_event\_info\_box () (in module indico.modules.events.abstracts.models.abstracts.AbstractEvent), 146  
 render\_registration\_event\_title\_placeholder () (in module indico.modules.events.abstracts.models.comments.AbstractComment), 149  
 render\_registration\_event\_review\_placeholder () (in module indico.modules.events.abstracts.models.reviews.AbstractReview), 156  
 render\_registration\_event\_submission\_placeholder () (in module indico.modules.events.contributions.models.contributions.Submission), 172  
 render\_registration\_event\_submission\_placeholder () (in module indico.modules.events.contributions.models.subcontributions.Subcontribution), 178  
 render\_registration\_event\_submission\_placeholder () (in module indico.modules.events.models.events.Event), 126  
 render\_registration\_event\_submission\_placeholder () (in module indico.modules.events.notes.models.notes.EventNoteRevision), 192  
 render\_registration\_event\_submission\_placeholder () (in module indico.modules.events.papers.models.comments.PaperReview), 194  
 render\_mode (indico.modules.events.papers.models.reviews.PaperReview), 200  
 render\_mode (indico.modules.events.papers.models.revisions.PaperRevision), 201  
 render\_mode (indico.modules.events.sessions.models.sessions.Session), 241  
 render\_mode (indico.modules.events.surveys.models.items.SurveyItem), 248  
 render\_mode (indico.modules.events.surveys.models.items.SurveyQuestion), 249  
 render\_mode (indico.modules.events.surveys.models.items.SurveySection), 250  
 render\_mode (indico.modules.events.surveys.models.items.SurveyText), 250  
 render\_mode (indico.modules.events.timetable.models.breaks.Break), 254  
 render\_mode (indico.modules.events.tracks.models.tracks.Track), 260  
 render\_payment\_form () (in module indico.modules.events.payment.plugins.PaymentPluginMixin), 209  
 render\_pdf () (in module indico.modules.events.contributions.util), 181  
 render\_price () (in module indico.modules.events.registration.models.registrations.RegistrationForm), 214  
 render\_price () (in module indico.modules.events.registration.models.registrations.RegistrationForm), 216  
 render\_price\_adjustment () (in module indico.modules.events.registration.models.registrations.RegistrationForm), 216

method), 214

render\_protection\_message() (in module `indico.web.forms.fields.IndicoProtectionField` method), 345

render\_session\_timetable() (in module `indico.modules.events.timetable.util`), 258

render\_session\_type\_row() (in module `indico.modules.events.sessions.util`), 245

render\_template\_func() (in module `indico.core.plugins.WPJinjaMixinPlugin` static method), 80

render\_transaction\_details() (in module `indico.modules.events.payment.plugins.PaymentPluginMixin` method), 209

renderer (`indico.modules.events.logs.models.entries.EventLogEntry` attribute), 187

repeat\_frequency (in module `indico.modules.rb.models.reservations.Reservation` attribute), 298

repeat\_interval (in module `indico.modules.rb.models.reservations.Reservation` attribute), 298

RepeatFrequency (class in module `indico.modules.rb.models.reservations`), 296

RepeatMapping (class in module `indico.modules.rb.models.reservations`), 296

repetition (`indico.modules.rb.models.reservations.Reservation` attribute), 298

reply\_to\_address (in module `indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate` attribute), 151

reply\_to\_address (in module `indico.modules.events.reminders.models.reminders.EventReminder` attribute), 235

Request (class in module `indico.modules.events.requests.models.requests`), 236

RequestDefinitionBase (class in module `indico.modules.events.requests.base`), 237

requested\_dt (`indico.modules.events.static.models.static.StaticState` attribute), 262

RequestState (class in module `indico.modules.events.requests.models.requests`), 237

require\_feature() (in module `indico.modules.events.features.util`), 181

require\_login (in module `indico.modules.events.registration.models.forms.RegistrationForm` attribute), 221

require\_user (`indico.modules.events.registration.models.forms.RegistrationForm` attribute), 221

require\_user (`indico.modules.events.surveys.models.surveys.Survey` attribute), 247

required (`indico.modules.events.agreements.placeholders.AgreementPlaceholder` attribute), 168

required (`indico.modules.events.registration.placeholders.invitations.InvitationPlaceholder` attribute), 232

RescheduleMode (class in module `indico.modules.events.timetable.reschedule`), 258

Rescheduler (class in module `indico.modules.events.timetable.reschedule`), 258

Reservation (class in module `indico.modules.rb.models.reservations`), 297

reservation\_id (in module `indico.modules.rb.models.reservation_edit_logs.ReservationEditLog` attribute), 300

reservation\_id (in module `indico.modules.rb.models.reservation_occurrences.ReservationOccurrence` attribute), 300

ReservationEditLog (class in module `indico.modules.rb.models.reservation_edit_logs`), 299

ReservationLink (class in module `indico.modules.rb.models.reservations`), 299

ReservationOccurrence (class in module `indico.modules.rb.models.reservation_occurrences`), 300

ReservationOccurrenceState (class in module `indico.modules.rb.models.reservation_occurrences`), 301

reservations (`indico.modules.events.models.events.Event` attribute), 226

reservations (`indico.modules.rb.models.rooms.Room` attribute), 292

reservations\_need\_confirmation (in module `indico.modules.rb.models.rooms.Room` attribute), 292

ReservationState (class in module `indico.modules.rb.models.reservations`), 299

reset() (`indico.modules.events.agreements.models.agreements.Agreement` method), 167

reset\_abstract\_state() (in module `indico.modules.events.abstracts.operations`), 157

reset\_approval() (in module `indico.modules.rb.models.reservations.Reservation` method), 298

reset\_client\_secret() (in module `indico.core.oauth.models.applications.OAuthApplication` method), 307

reset\_paper\_state() (in module `indico.modules.events.papers.operations`), 203

reset\_signing\_secret() (in module `indico.modules.users.models.users.User` method), 221

[reset\\_state\(\)](#) (in [attribute](#)), 134  
[dico.modules.events.abstracts.models.abstracts.AbstractRevisionAttr](#) (in-  
[method](#)), 146  
[reset\\_state\(\)](#) (in [attribute](#)), 200  
[dico.modules.events.papers.models.papers.Paper](#) [revision\\_count](#) (in-  
[method](#)), 197  
[dico.modules.events.papers.models.papers.Paper](#)  
[resolve\\_title\(\)](#) (in [module](#) in [attribute](#)), 197  
[dico.modules.vc.util](#)), 314  
[restore\(\)](#) ([indico.modules.events.models.events.Event](#)  
[method](#)), 126  
[restored](#) (in [module](#) [indico.core.signals.event](#)), 86  
[RESULTS\\_PER\\_PAGE](#) (in [attribute](#)), 70  
[dico.modules.search.base.IndicoSearchProvider](#)  
[ResultSchemaBase](#) (class in [in-  
\[dico.modules.search.result\\\_schemas\]\(#\)\), 71  
\[review\]\(#\) \(\[indico.modules.events.abstracts.models.review\\\_ratings.AbstractReviewRating\]\(#\)  
\[attribute\]\(#\)\), 154  
\[review\]\(#\) \(\[indico.modules.events.papers.models.review\\\_ratings.PaperReviewRating\]\(#\)  
\[attribute\]\(#\)\), 198  
\[review\\\_class\]\(#\) \(\[indico.modules.events.abstracts.models.review\\\_ratings.AbstractReviewRating\]\(#\) \(in-  
\[attribute\]\(#\)\), 154  
\[dico.modules.events.models.reviews.ProposalMixin\]\(#\)  
\[review\\\_class\]\(#\) \(\[indico.modules.events.papers.models.review\\\_ratings.PaperReviewRating\]\(#\)  
\[attribute\]\(#\)\), 198  
\[revisions\\\_enabled\]\(#\) \(in-  
\[indico.modules.events.abstracts.models.abstracts.AbstractRevisionMixin\]\(#\)  
\[review\\\_id\]\(#\) \(\[indico.modules.events.abstracts.models.review\\\_ratings.AbstractReviewRating\]\(#\)  
\[attribute\]\(#\)\), 154  
\[review\\\_id\]\(#\) \(\[indico.modules.events.papers.models.review\\\_ratings.PaperReviewRating\]\(#\) \(in-  
\[attribute\]\(#\)\), 198  
\[dico.modules.events.papers.models.papers.Paper\]\(#\)  
\[reviewed\\\_for\]\(#\) \(\[indico.modules.events.abstracts.models.abstracts.EditTrackModule\]\(#\)  
\[attribute\]\(#\)\), 148  
\[rewrite\\\_css\\\_urls\\(\\)\]\(#\) \(in \[module\]\(#\) in \[in-  
\\[dico.modules.events.static.util\\]\\(#\\)\\), 263  
\\[reviewed\\\\_for\\\\_tracks\\]\\(#\\) \\(in \\[attribute\\]\\(#\\)\\), 146  
\\[dico.modules.events.abstracts.models.abstracts.AbstractRevisionMixin\\]\\(#\\)  
\\[reviewers\\]\\(#\\) \\(\\[indico.modules.events.abstracts.models.review\\\\_ratings.AbstractReviewRating\\]\\(#\\) \\(class in \\[in-  
\\\[dico.modules.events.static.util\\\]\\\(#\\\)\\\), 263  
\\\[reviewers\\\]\\\(#\\\) \\\(\\\[indico.modules.events.papers.models.reviews.PaperCommentsVisibility\\\]\\\(#\\\) \\\(class in \\\[in-  
\\\\[dico.modules.events.management.controllers\\\\]\\\\(#\\\\)\\\\),  
\\\\[reviewing\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.events.logs.models.entries.EventLogRedline\\\\]\\\\(#\\\\)  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 188  
\\\\[Room\\\\]\\\\(#\\\\) \\\\(class in \\\\[indico.modules.rb.models.rooms\\\\]\\\\(#\\\\)\\\\), 289  
\\\\[reviewing\\\\\\_instructions\\\\]\\\\(#\\\\) \\\\(in \\\\[room\\\\\\_id\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.rb.models.blocked\\\\\\_rooms.BlockedRoom\\\\]\\\\(#\\\\)  
\\\\[dico.modules.events.abstracts.models.call\\\\\\_for\\\\\\_abstracts.CallForAbstracts\\\\]\\\\(#\\\\)\\\\), 284  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 148  
\\\\[room\\\\\\_id\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.rb.models.reservations.Reservation\\\\]\\\\(#\\\\)  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 298  
\\\\[reviewing\\\\\\_state\\\\]\\\\(#\\\\) \\\\(in \\\\[abstract\\\\\\_id\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.rb.models.room\\\\\\_attributes.RoomAttributeAssoc\\\\]\\\\(#\\\\)  
\\\\[dico.modules.events.abstracts.models.abstracts.AbstractRevisionMixin\\\\]\\\\(#\\\\) \\\\(in \\\\[indico.modules.rb.models.room\\\\\\_bookable\\\\\\_hours.BookableHour\\\\]\\\\(#\\\\)  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 293  
\\\\[revision\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.events.models.reviews.ProposalRevisionMixin\\\\]\\\\(#\\\\) \\\\(in \\\\[indico.modules.rb.models.room\\\\\\_nonbookable\\\\\\_periods.NonBook\\\\]\\\\(#\\\\)  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 200  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 293  
\\\\[revision\\\\\\_attr\\\\]\\\\(#\\\\) \\\\(in \\\\[room\\\\\\_name\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.events.contributions.models.subcontributions\\\\]\\\\(#\\\\)  
\\\\[dico.modules.events.abstracts.models.reviews.AbstractRevisionMixin\\\\]\\\\(#\\\\) \\\\(in \\\\[indico.modules.events.contributions.models.subcontributions\\\\]\\\\(#\\\\)  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 178  
\\\\[attribute\\\\]\\\\(#\\\\)\\\\), 156  
\\\\[room\\\\\\_name\\\\]\\\\(#\\\\) \\\\(\\\\[indico.modules.search.result\\\\\\_schemas.LocationResultSchema\\\\]\\\\(#\\\\)  
\\\\[revision\\\\\\_attr\\\\]\\\\(#\\\\) \\\\(in \\\\[attribute\\\\]\\\\(#\\\\)\\\\), 73  
\\\\[dico.modules.events.models.reviews.ProposalRevisionMixin\\\\]\\\\(#\\\\) \\\\(in \\\\[attribute\\\\]\\\\(#\\\\)\\\\), 73  
\\\\[name\\\\\\_format\\\\]\\\\(#\\\\) \\\\(in \\\\[attribute\\\\]\\\\(#\\\\)\\\\), 73\\\]\\\(#\\\)\\]\\(#\\)\]\(#\)](#)

*dico.modules.rb.models.locations.Location* attribute), 296

*RoomAttribute* (class in *indico.modules.rb.models.room\_attributes*), 292

*RoomAttributeAssociation* (class in *indico.modules.rb.models.room\_attributes*), 292

*rooms* (*indico.modules.rb.models.locations.Location* attribute), 296

*ROUTE\_OLD\_URLS* (built-in variable), 56

*rules* (*indico.modules.events.abstracts.models.email\_templates.AbstractEmailTemplate* attribute), 151

*run()* (*indico.modules.events.timetable.reschedule.Rescheduler* method), 259

*running* (*indico.modules.events.static.models.static.StaticSiteState* attribute), 263

## S

*safe\_last\_login\_dt* (in *indico.modules.auth.models.identities.Identity* attribute), 303

*save()* (*indico.modules.users.ext.ExtraUserPreferences* method), 281

*save\_identity\_info()* (in module *indico.modules.auth.util*), 304

*save\_submitted\_survey\_to\_session()* (in module *indico.modules.events.surveys.util*), 253

*schedule* (*indico.modules.events.abstracts.settings.BOASortField* attribute), 165

*schedule()* (*indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstract* method), 148

*schedule()* (*indico.modules.events.papers.models.call\_for\_papers.CallForPapers* method), 193

*schedule\_board\_number* (in *indico.modules.events.abstracts.settings.BOASortField* attribute), 165

*schedule\_cfa()* (in module *indico.modules.events.abstracts.operations*), 157

*schedule\_cfp()* (in module *indico.modules.events.papers.operations*), 203

*schedule\_contribution()* (in module *indico.modules.events.timetable.operations*), 256

*scheduled\_dt* (*indico.modules.events.reminders.models.reminders.EventReminder* attribute), 235

*scheduled\_notes* (in *indico.modules.events.models.events.Event* attribute), 126

*SCHEDULED\_TASK\_OVERRIDE* (built-in variable), 49

*schema\_post\_dump* (in module *indico.core.signals.plugin*), 88

*schema\_post\_load* (in module *indico.core.signals.plugin*), 89

*schema\_pre\_load* (in module *indico.core.signals.plugin*), 89

*schema* (*indico.modules.events.models.references.ReferenceType* attribute), 132

*scope* (*indico.core.oauth.models.tokens.OAuth2AuthorizationCode* attribute), 308

*scopes* (*indico.core.oauth.models.applications.OAuthApplicationUserLineItem* attribute), 170

*scopes* (*indico.core.oauth.models.tokens.TokenModelBase* attribute), 310

*score* (*indico.modules.events.abstracts.models.abstracts.Abstract* attribute), 146

*score* (*indico.modules.events.abstracts.models.reviews.AbstractReview* attribute), 156

*score* (*indico.modules.events.models.reviews.ProposalReviewMixin* attribute), 134

*score* (*indico.modules.events.papers.models.reviews.PaperReview* attribute), 200

*score* (*indico.modules.users.models.suggestions.SuggestedCategory* attribute), 277

*search()* (*indico.modules.groups.core.GroupProxy* class method), 311

*search()* (*indico.modules.search.base.IndicoSearchProvider* method), 70

*search\_data* (*indico.modules.events.registration.models.registrations.Registration* attribute), 216

*search\_payload* (in *indico.modules.events.abstracts.fields.AbstractField* attribute), 328

*search\_payload* (in *indico.modules.events.abstracts.fields.AbstractField* attribute), 328

*search\_users()* (in module *indico.modules.users.util*), 280

*SearchOption* (class in *indico.modules.search.base*), 70

*SearchOptions* (class in *indico.modules.search.base*), 70

*SearchTarget* (class in *indico.modules.search.base*), 70

*secondary* (*indico.modules.events.contributions.models.persons.AuthorTopic* attribute), 174

*secondary\_authors* (in *indico.modules.events.models.persons.AuthorsSpeakersMixin* attribute), 128

*secondary\_emails* (in *indico.modules.users.models.users.User* attribute), 275

*secondary\_local\_identities* (in *indico.modules.users.models.users.User* attribute), 275



SECRET\_KEY (built-in variable), 54

section (indico.modules.events.registration.models.items.RegistrationFormType\_chain() (in module indico.modules.categories.serialize), 269

attribute), 224

section (indico.modules.events.surveys.models.items.SurveyFormType\_category\_role() (in module indico.modules.categories.util), 269

attribute), 248

section\_pd (indico.modules.events.registration.models.items.RegistrationFormType\_pre\_bookings() (in module indico.modules.rb.util), 301

attribute), 224

sections (in module indico.core.signals.menu), 88

sections (indico.modules.events.registration.models.forms.RegistrationForm module indico.modules.events.contributions.util), 181

attribute), 221

sections (indico.modules.events.surveys.models.surveys.SurveyForm module indico.modules.events.contributions.util), 181

attribute), 247

sections\_with\_answered\_fields (in module indico.modules.events.registration.models.registration\_event\_for\_ical() (in module indico.modules.events.util), 141

attribute), 214

send() (indico.modules.events.reminders.models.reminders.EventReminder\_event\_for\_json\_ld() (in module indico.modules.events.util), 141

method), 235

send() (indico.modules.events.requests.base.RequestDefinitionBase\_serialize\_event\_person() (in module indico.modules.events.util), 141

class method), 239

send\_avatar() (in module indico.modules.users.util), 280

serialize\_group() (in module indico.modules.groups.util), 311

send\_default\_avatar() (in module indico.modules.users.util), 280

serialize\_ip\_network\_group() (in module indico.modules.networks.util), 326

send\_new\_agreements() (in module indico.modules.events.agreements.util), 167

serialize\_log\_entry() (in module indico.modules.events.logs.util), 188

send\_start\_notification() (in module indico.modules.events.surveys.models.surveys.Survey\_serialize\_nonbookable\_periods() (in module indico.modules.rb.util), 302

method), 247

serialize\_occurrences() (in module indico.modules.rb.util), 302

send\_submission\_notification() (in module indico.modules.events.surveys.models.surveys.Survey\_serialize\_person\_for\_json\_ld() (in module indico.modules.events.util), 141

method), 247

send\_to\_participants (in module indico.modules.events.reminders.models.reminders.EventReminder\_serialize\_person\_link() (in module indico.modules.events.util), 141

attribute), 235

serialize\_registration\_form() (in module indico.modules.events.registration.util), 229

send\_to\_speakers (in module indico.modules.events.reminders.models.reminders.EventReminder\_serialize\_session\_for\_ical() (in module indico.modules.events.sessions.util), 245

attribute), 235

sent\_dt (indico.modules.events.abstracts.models.email\_logs.AbstractEmailLogEmail\_log\_entry\_hours() (in module indico.modules.rb.util), 302

attribute), 150

serialize\_user() (in module indico.modules.users.util), 280

SENTRY\_DSN (built-in variable), 54

SENTRY\_LOGGING\_LEVEL (built-in variable), 54

separator (indico.modules.events.layout.models.menu.MenuFormType\_indico.modules.events.models.events.Event attribute), 185

serialize\_availability() (in module indico.modules.rb.util), 301

series\_id (indico.modules.events.models.events.Event attribute), 126

serialize\_blockings() (in module indico.modules.rb.util), 301

service\_name (indico.modules.vc.plugins.VCPluginMixin attribute), 316

serialize\_booking\_details() (in module indico.modules.rb.util), 301

Session (class in module indico.modules.events.sessions.models.sessions), 239

serialize\_categories\_ical() (in module indico.modules.categories.serialize), 269

session (indico.modules.attachments.models.folders.AttachmentFolder attribute), 285

serialize\_category() (in module indico.modules.categories.serialize), 269

session (indico.modules.events.contributions.models.contributions.Contribution attribute), 172

serialize\_category\_atom() (in module indico.modules.categories.serialize), 269

session (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 178	session (indico.modules.events.abstracts.settings.BOASortField attribute), 165
session (indico.modules.events.notes.models.notes.EventNote attribute), 191	session_coordinator_priv_enabled() (in module indico.modules.events.sessions.util), 245
session (indico.modules.events.sessions.models.sessions.Session attribute), 241	session_deleted (in module indico.core.signals.event), 86
session (indico.modules.rb.models.reservations.ReservationLink attribute), 299	session_id (indico.modules.attachments.models.folders.AttachmentFolder attribute), 285
session_block (indico.modules.attachments.models.folders.AttachmentFolder attribute), 285	session_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 172
session_block (indico.modules.events.contributions.models.contributions.Contribution attribute), 172	session_id (indico.modules.events.notes.models.notes.EventNote attribute), 191
session_block (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 243	session_id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 244
session_block (indico.modules.events.notes.models.notes.EventNote attribute), 191	session_id (indico.modules.rb.models.reservations.ReservationLink attribute), 299
session_block (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 255	SESSION_LIFETIME (built-in variable), 55
SESSION_BLOCK (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 256	session_type (in module indico.modules.events.abstracts.settings.BOASortField attribute), 165
session_block (indico.modules.rb.models.reservations.ReservationLink attribute), 299	session_siblings (in module indico.modules.events.timetable.models.entries.TimetableEntry attribute), 255
session_block_count (indico.modules.events.models.events.Event attribute), 127	session_title (in module indico.modules.events.abstracts.settings.BOASortField attribute), 165
session_block_deleted (in module indico.core.signals.event), 86	session_updated (in module indico.core.signals.event), 86
session_block_id (indico.modules.attachments.models.folders.AttachmentFolder attribute), 285	SessionBlock (class in indico.modules.events.sessions.models.blocks), 241
session_block_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 172	SessionPersonLink (class in indico.modules.events.sessions.models.persons), 243
session_block_id (indico.modules.events.notes.models.notes.EventNote attribute), 191	SessionBlockPersonLinkListField (class in indico.modules.events.sessions.fields), 333
session_block_id (indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 243	SessionListToPDF (class in indico.modules.events.sessions.util), 244
session_block_id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 244	SessionPrincipal (class in indico.modules.events.sessions.models.principals), 243
session_block_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 255	set() (indico.modules.categories.settings.CategorySettingsProxy method), 270
session_block_id (indico.modules.rb.models.reservations.ReservationLink attribute), 299	set() (indico.modules.events.settings.EventACLProxy method), 142, 205
session_block_id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 313	set() (indico.modules.events.settings.EventSettingsProxy method), 143, 206
session_block_updated (in module indico.core.signals.event), 86	set() (indico.modules.users.models.settings.UserSettingsProxy method), 278
	set_attribute_value() (in module indico.modules.events.settings.EventACLProxy method), 142, 205

[dico.modules.rb.models.rooms.Room method](#)), [settings\\_form](#) (in-  
[292](#) [dico.modules.events.payment.plugins.PaymentPluginMixin](#)  
[set\\_custom\\_field\(\)](#) (in- [attribute](#)), [209](#)  
[dico.modules.events.contributions.models.contributions.CustomFieldsMixin](#) (in-  
[method](#)), [172](#) [dico.modules.vc.plugins.VCPluginMixin](#)  
[set\\_custom\\_fields\(\)](#) (in module in- [attribute](#)), [316](#)  
[dico.modules.events.util](#)), [141](#) [settings\\_form\\_field\\_opts](#) (in-  
[set\\_deadline\(\)](#) (in module in- [dico.core.plugins.IndicoPlugin attribute](#)),  
[dico.modules.events.papers.operations](#)), [78](#)  
[203](#) [shell\\_context](#) (in module in-  
[set\\_feature\\_enabled\(\)](#) (in module in- [dico.core.signals.plugin](#)), [89](#)  
[dico.modules.events.features.util](#)), [181](#) [shift\\_following\\_entries\(\)](#) (in module in-  
[set\\_multi\(\)](#) ([indico.modules.categories.settings.CategorySettingsProxy](#) [dico.modules.events.timetable.util](#)), [258](#)  
[method](#)), [271](#) [short\\_external\\_url](#) (in-  
[set\\_multi\(\)](#) ([indico.modules.events.settings.EventSettingsProxy](#) [dico.modules.events.models.events.Event](#)  
[method](#)), [144](#), [206](#) [attribute](#)), [127](#)  
[set\\_multi\(\)](#) ([indico.modules.users.models.settings.UsersSettingsProxy](#) (in- [dico.modules.events.tracks.models.tracks.Track](#)  
[method](#)), [278](#) [attribute](#)), [260](#)  
[set\\_participant\\_list\\_columns\(\)](#) (in- [short\\_title\\_with\\_group](#) (in-  
[dico.modules.events.registration.settings.RegistrationSettingsProxy](#) [dico.modules.events.tracks.models.tracks.Track](#)  
[method](#)), [232](#) [attribute](#)), [260](#)  
[set\\_participant\\_list\\_form\\_ids\(\)](#) (in- [short\\_url](#) ([indico.modules.events.models.events.Event](#)  
[dico.modules.events.registration.settings.RegistrationSettingsProxy](#) [attribute](#)), [127](#)  
[method](#)), [232](#) [should\\_show\\_draft\\_warning\(\)](#) (in module in-  
[set\\_reviewing\\_state\(\)](#) (in module in- [dico.modules.events.util](#)), [141](#)  
[dico.modules.events.papers.operations](#)), [203](#) [show](#) ([indico.modules.vc.models.vc\\_rooms.VCRoomEventAssociation](#)  
[203](#) [attribute](#)), [313](#)  
[set\\_reviewing\\_state\(\)](#) (in- [show\\_links](#) ([indico.modules.events.models.series.EventSeries](#)  
[dico.modules.events.papers.models.call\\_for\\_papers.CallForPapers](#) [attribute](#)), [135](#)  
[method](#)), [193](#) [show\\_sequence\\_in\\_title](#) (in-  
[set\\_user\\_avatar\(\)](#) (in module in- [dico.modules.events.models.series.EventSeries](#)  
[dico.modules.users.util](#)), [280](#) [attribute](#)), [135](#)  
[settings](#) ([indico.core.plugins.IndicoPlugin attribute](#)), [78](#) [siblings](#) ([indico.modules.events.timetable.models.entries.TimetableEntry](#)  
[78](#) [attribute](#)), [255](#)  
[settings](#) ([indico.modules.auth.models.registration\\_requests.RegistrationRequest](#) (in-  
[attribute](#)), [303](#) [dico.modules.events.timetable.models.entries.TimetableEntry](#)  
[settings](#) ([indico.modules.events.settings.ThemeSettingsProxy](#) [attribute](#)), [255](#)  
[attribute](#)), [144](#), [206](#) [sidemenu](#) (in module [indico.core.signals.event](#)), [86](#)  
[settings](#) ([indico.modules.users.models.users.User attribute](#)), [275](#) [signed\\_dt](#) ([indico.modules.events.agreements.models.agreements.Agreement](#)  
[attribute](#)), [167](#)  
[settings\\_backref\\_name](#) (in- [signed\\_from\\_ip](#) (in-  
[dico.modules.events.models.settings.EventSetting](#) [dico.modules.events.agreements.models.agreements.Agreement](#)  
[attribute](#)), [135](#) [attribute](#)), [167](#)  
[settings\\_backref\\_name](#) (in- [signed\\_on\\_behalf](#) (in-  
[dico.modules.events.models.settings.EventSettingPrincipal](#) [dico.modules.events.agreements.models.agreements.Agreement](#)  
[attribute](#)), [136](#) [attribute](#)), [167](#)  
[settings\\_backref\\_name](#) (in- [signing\\_secret](#) (in-  
[dico.modules.events.models.settings.EventSettingsMixin](#) [dico.modules.users.models.users.User](#) [attribute](#)),  
[attribute](#)), [136](#) [275](#)  
[settings\\_converters](#) (in- [SimpleRenderer](#) (class in in-  
[dico.core.plugins.IndicoPlugin attribute](#)), [78](#) [dico.modules.events.logs.renderers](#)), [189](#)  
[78](#) [site](#) ([indico.modules.rb.models.rooms.Room attribute](#)),  
[settings\\_form](#) ([indico.core.plugins.IndicoPlugin attribute](#)), [78](#) [292](#)  
[78](#) [size](#) ([indico.modules.attachments.models.attachments.AttachmentFile](#)

attribute), 283  
 size (indico.modules.designer.models.images.DesignerImageFile attribute), 317  
 size (indico.modules.events.abstracts.models.files.AbstractFile attribute), 152  
 size (indico.modules.events.layout.models.images.ImageFile attribute), 182  
 size (indico.modules.events.papers.models.files.PaperFile attribute), 196  
 size (indico.modules.events.papers.models.templates.PaperTemplate attribute), 203  
 size (indico.modules.events.registration.models.registrations.Registration attribute), 216  
 size (indico.modules.events.static.models.static.StaticSite attribute), 262  
 skip\_moderation (in-  
   dico.modules.events.registration.models.invitations.Registration attribute), 222  
 slug (indico.modules.events.contributions.models.contributions.Contribution attribute), 172  
 slug (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 178  
 slug (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 243  
 slug (indico.modules.news.models.news.NewsItem attribute), 327  
 SMTP\_LOGIN (built-in variable), 53  
 SMTP\_PASSWORD (built-in variable), 53  
 SMTP\_SERVER (built-in variable), 53  
 SMTP\_TIMEOUT (built-in variable), 53  
 SMTP\_USE\_CELERY (built-in variable), 52  
 SMTP\_USE\_TLS (built-in variable), 53  
 sort\_contribs() (in module in-  
   dico.modules.events.contributions.util), 181  
 sort\_reviewing\_questions() (in module in-  
   dico.modules.events.operations), 138  
 source (indico.modules.events.notes.models.notes.EventNote attribute), 192  
 speaker (indico.modules.events.abstracts.settings.BOASpeaker attribute), 165  
 speakers (indico.modules.events.abstracts.settings.BOASpeakers attribute), 165  
 speakers (indico.modules.events.abstracts.settings.SubmissionRights attribute), 165  
 speakers (indico.modules.events.contributions.models.subcontributions.SubContributors attribute), 178  
 speakers (indico.modules.events.models.persons.AuthorsSpeakers attribute), 128  
 split\_data (indico.web.forms.fields.RelativeDeltaField attribute), 346  
 spotlight\_file (in-  
   dico.modules.events.papers.models.revisions.PaperRevision attribute), 201  
 sprite\_position (in-  
   dico.modules.rb.models.rooms.Room attribute), 292  
 SQLALCHEMY\_DATABASE\_URI (built-in variable), 51  
 SQLALCHEMY\_POOL\_RECYCLE (built-in variable), 51  
 SQLALCHEMY\_POOL\_SIZE (built-in variable), 51  
 SQLALCHEMY\_POOL\_TIMEOUT (built-in variable), 51  
 standard (indico.modules.users.models.users.ProfilePictureSource attribute), 272  
 start\_date (indico.modules.rb.models.blockings.Blocking attribute), 294  
 start\_dt (indico.modules.events.abstracts.models.call\_for\_abstracts.CallForAbstracts attribute), 148  
 start\_dt (indico.modules.events.contributions.models.contributions.Contribution attribute), 172  
 start\_dt (indico.modules.events.models.events.Event attribute), 127  
 start\_dt (indico.modules.events.papers.models.call\_for\_papers.CallForPapers attribute), 194  
 start\_dt (indico.modules.events.registration.models.forms.RegistrationForm attribute), 221  
 start\_dt (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 243  
 start\_dt (indico.modules.events.sessions.models.sessions.Session attribute), 241  
 start\_dt (indico.modules.events.surveys.models.surveys.Survey attribute), 247  
 start\_dt (indico.modules.events.timetable.models.breaks.Break attribute), 254  
 start\_dt (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 255  
 start\_dt (indico.modules.rb.models.reservation\_occurrences.ReservationOccurrences attribute), 300  
 start\_dt (indico.modules.rb.models.reservations.Reservation attribute), 298  
 start\_dt (indico.modules.rb.models.room\_nonbookable\_periods.NonBookablePeriod attribute), 293  
 start\_dt (indico.modules.search.result\_schemas.ContributionResultSchema attribute), 72  
 start\_dt (indico.modules.search.result\_schemas.EventResultSchema attribute), 71  
 start\_dt\_display (in-  
   dico.modules.events.contributions.models.contributions.Contribution attribute), 172  
 start\_dt\_override (in-  
   dico.modules.events.models.events.Event attribute), 127  
 start\_dt\_poster (in-  
   dico.modules.events.contributions.models.contributions.Contribution attribute), 172



[attribute](#)), 172  
[start\\_notification\\_emails](#) (in- [dico.modules.events.surveys.models.surveys.SurveyStatsBase](#) (class in [dico.modules.events.surveys.models.surveys.SurveyStatsBase](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 247  
[start\\_notification\\_recipients](#) (in- [dico.modules.events.surveys.models.surveys.Survey](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 247  
[start\\_notification\\_sent](#) (in- [dico.modules.events.surveys.models.surveys.Survey](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 247  
[start\\_time](#) ([indico.modules.rb.models.room\\_bookable\\_hours.BookableHours](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 293  
[starts\\_between\(\)](#) (in- [dico.modules.events.models.events.Event](#) (class in [dico.modules.events.registration.stats](#)), 234  
[method](#)), 127  
[state](#) ([indico.modules.events.abstracts.models.abstracts.AbstractEvent](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 146  
[state](#) ([indico.modules.events.agreements.models.agreements.Agreement](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 167  
[state](#) ([indico.modules.events.papers.models.papers.Paper](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 197  
[state](#) ([indico.modules.events.papers.models.revisions.PaperRevision](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 201  
[state](#) ([indico.modules.events.registration.models.invitations.RegistrationInvitation](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 222  
[state](#) ([indico.modules.events.registration.models.registrations.Registration](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 214  
[state](#) ([indico.modules.events.requests.models.requests.Request](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 237  
[state](#) ([indico.modules.events.static.models.static.StaticSite](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 262  
[state](#) ([indico.modules.events.surveys.models.surveys.Survey](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 247  
[State](#) ([indico.modules.rb.models.blocked\\_rooms.BlockedRoom](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 294  
[state](#) ([indico.modules.rb.models.blocked\\_rooms.BlockedRoom](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 294  
[state](#) ([indico.modules.rb.models.reservation\\_occurrences.ReservationOccurrence](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 301  
[state](#) ([indico.modules.rb.models.reservations.Reservation](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 298  
[state\\_name](#) ([indico.modules.rb.models.blocked\\_rooms.BlockedRoom](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 294  
[STATIC\\_FILE\\_METHOD](#) (built-in variable), 56  
[static\\_items](#) ([indico.modules.events.util.ListGeneratorBase](#) (class in [dico.modules.events.registration.stats](#)), 234  
[attribute](#)), 139  
[STATIC\\_SITE\\_STORAGE](#) (built-in variable), 55  
[StaticListLink](#) (class in [dico.modules.events.registration.stats](#)), 234  
[dico.modules.events.models.static\\_list\\_links](#)), 136  
[StaticSite](#) (class in [dico.modules.events.registration.stats](#)), 234  
[dico.modules.events.static.models.static](#)), 262  
[StaticSiteState](#) (class in [dico.modules.events.registration.stats](#)), 234

[storage\\_file\\_id](#) (in- [subcontribution\\_id](#) (in- [dico.modules.events.registration.models.registrations.RegistrationModule](#) [dico.modules.events.contributions.models.references.SubContributionReference](#) [attribute](#)), 216 [attribute](#)), 177  
[storage\\_file\\_id](#) (in- [subcontribution\\_id](#) (in- [dico.modules.events.static.models.static.StaticSite](#) [dico.modules.events.notes.models.notes.EventNote](#) [attribute](#)), 262 [attribute](#)), 192  
[store\\_configuration\(\)](#) (in- [subcontribution\\_id](#) (in- [dico.modules.events.util.ListGeneratorBase](#) [dico.modules.rb.models.reservations.ReservationLink](#) [method](#)), 139 [attribute](#)), 299  
[stored\\_file\\_class](#) (in- [subcontribution\\_id](#) (in- [dico.modules.attachments.models.attachments.Attachment](#) [dico.modules.search.result\\_schemas.AttachmentResultSchema](#) [attribute](#)), 282 [attribute](#)), 72  
[stored\\_file\\_fkey](#) (in- [subcontribution\\_id](#) (in- [dico.modules.attachments.models.attachments.Attachment](#) [dico.modules.search.result\\_schemas.EventNoteResultSchema](#) [attribute](#)), 282 [attribute](#)), 73  
[stored\\_file\\_table](#) (in- [subcontribution\\_id](#) (in- [dico.modules.attachments.models.attachments.Attachment](#) [dico.modules.search.result\\_schemas.SubContributionResultSchema](#) [attribute](#)), 282 [attribute](#)), 72  
[STRICT\\_LATEX](#) (built-in variable), 54 [subcontribution\\_updated](#) (in module in- [dico.core.signals.event](#)), 87  
[strict\\_settings](#) (indico.core.plugins.IndicoPlugin [SubContributionPersonLink](#) (class in in- [attribute](#)), 78 [dico.modules.events.contributions.models.persons](#)), 175  
[stylesheet](#) (indico.modules.events.models.events.Event [attribute](#)), 127  
[stylesheet\\_metadata](#) (in- [SubContributionPersonLinkListField](#) (class in [dico.modules.events.models.events.Event](#) [indico.modules.events.contributions.fields](#)), 127 [attribute](#)), 331  
[SubContribution](#) (class in in- [SubContributionReference](#) (class in in- [dico.modules.events.contributions.models.subcontributions](#)) [dico.modules.events.contributions.models.references](#)), 177 [attribute](#)), 177  
[subcontribution](#) (in- [SubContributionResultSchema](#) (class in in- [dico.modules.attachments.models.folders.AttachmentFolder](#) [dico.modules.search.result\\_schemas](#)), 72 [attribute](#)), 285  
[subcontribution](#) (in- [subcontributions](#) (in- [dico.modules.events.contributions.models.contributions.Contribution](#) [attribute](#)), 172  
[subcontribution](#) (in- [dico.modules.events.notes.models.notes.EventNote](#) [attribute](#)), 191  
[subcontribution](#) (in- [subject](#) (indico.modules.events.abstracts.models.email\_logs.AbstractEmailLog) [attribute](#)), 150  
[subcontribution](#) (in- [dico.modules.rb.models.reservations.ReservationLink](#) [subject](#) (indico.modules.events.abstracts.models.email\_templates.AbstractEmailTemplate) [attribute](#)), 299 [attribute](#)), 151  
[subcontribution](#) (in- [submission\\_comment](#) (in- [dico.modules.search.base.SearchTarget](#) [dico.modules.events.abstracts.models.abstracts.AbstractEvent](#) [attribute](#)), 70 [attribute](#)), 146  
[subcontribution\\_count](#) (in- [submission\\_id](#) (in- [dico.modules.events.contributions.models.contributions.Contribution](#) [dico.modules.events.surveys.models.submissions.SurveyAnswer](#) [attribute](#)), 172 [attribute](#)), 251  
[subcontribution\\_created](#) (in module in- [submission\\_instructions](#) (in- [dico.core.signals.event](#)), 87 [dico.modules.events.abstracts.models.call\\_for\\_abstracts.CallForAbstract](#) [attribute](#)), 148  
[subcontribution\\_deleted](#) (in module in- [submission\\_limit](#) (in- [dico.core.signals.event](#)), 87 [dico.modules.events.surveys.models.surveys.Survey](#) [attribute](#)), 247  
[subcontribution\\_id](#) (in- [SubmissionRightsType](#) (class in in- [dico.modules.attachments.models.folders.AttachmentFolder](#) [dico.modules.events.abstracts.settings](#)), 165 [attribute](#)), 285  
[subcontribution\\_id](#) (in- [dico.modules.events.contributions.models.persons.SubContributionPersonLink](#) (class in in- [dico.modules.events.surveys.models.surveys.Survey](#) [attribute](#)), 175 [attribute](#)), 247

submitted (*indico.modules.events.abstracts.models.abstracts.AbstractSubmission* attribute), 148

submitted (*indico.modules.events.papers.models.revisions.PaperRevision* attribute), 202

submitted (*indico.modules.events.static.models.static.StaticSiteState* attribute), 263

submitted\_contrib\_type (in- *indico.modules.events.payment.models.transactions.Transaction* attribute), 208

submitted\_contrib\_type\_id (in- *indico.modules.users.models.users.User* attribute), 275

submitted\_dt (*indico.modules.events.abstracts.models.abstracts.AbstractSubmission* attribute), 147

submitted\_dt (*indico.modules.events.papers.models.revisions.PaperRevision* attribute), 201

submitted\_dt (*indico.modules.events.registration.models.registrations.Registration* attribute), 215

submitted\_dt (*indico.modules.events.logs.models.entries.EventLogEntry* attribute), 187

submitted\_dt (*indico.modules.events.surveys.models.submissions.SurveySubmission* attribute), 251

submitted\_for\_tracks (in- *indico.modules.events.registration.models.registrations.Registration* attribute), 215

submitter (*indico.modules.events.abstracts.models.abstracts.AbstractSubmission* attribute), 147

submitter (*indico.modules.events.abstracts.settings.AllowEditingType* attribute), 164

submitter (*indico.modules.events.abstracts.settings.BOACorrespondingAuthorType* attribute), 165

submitter (*indico.modules.events.papers.models.revisions.PaperRevision* attribute), 201

submitter\_all (in- *indico.modules.events.surveys.models.items.SurveyItem* attribute), 165

submitter\_authors (in- *indico.modules.events.surveys.models.items.SurveyQuestion* attribute), 165

submitter\_id (*indico.modules.events.abstracts.models.abstracts.AbstractSubmission* attribute), 147

submitter\_id (*indico.modules.events.papers.models.revisions.PaperRevision* attribute), 202

submitter\_primary (in- *indico.modules.events.surveys.models.submissions.SurveySubmission* attribute), 165

SubmitterFirstNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 161

SubmitterLastNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 162

SubmitterNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 161

submitters (*indico.modules.events.contributions.models.contributions.Contribution* attribute), 172

SubmitterTitlePlaceholder (class in *indico.modules.events.surveys.models.items*), 161

- 249
- SurveyState (class in *indico.modules.events.surveys.models.surveys*), 247
- SurveySubmission (class in *indico.modules.events.surveys.models.submissions*), 251
- SurveyText (class in *indico.modules.events.surveys.models.items*), 250
- swap\_timetable\_entry() (in module *indico.modules.events.timetable.operations*), 256
- sync\_state() (*indico.modules.events.registration.models.registration.Registration* method), 215
- sync\_user() (*indico.modules.events.models.persons.EventPerson* method), 129
- syncable\_fields (in module *indico.modules.users.models.users*), 276
- synced\_fields (in *indico.modules.users.models.users.User* attribute), 275
- synced\_values (in *indico.modules.users.models.users.User* attribute), 275
- synchronize\_data() (in *indico.modules.users.models.users.User* method), 275
- system\_app\_type (in *indico.core.oauth.models.applications.OAuthApplication* attribute), 307
- SYSTEM\_NOTICES\_URL (built-in variable), 52
- SystemAppType (class in *indico.core.oauth.models.applications*), 307
- T**
- TargetAbstractIDPlaceholder (class in *indico.modules.events.abstracts.placeholders*), 162
- TargetAbstractTitlePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 162
- TargetSubmitterFirstNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 163
- TargetSubmitterLastNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 163
- TargetSubmitterNamePlaceholder (class in *indico.modules.events.abstracts.placeholders*), 163
- telephone (*indico.modules.rb.models.rooms.Room* attribute), 292
- TEMP\_DIR (built-in variable), 53
- TEMPATE (*indico.modules.attachments.preview.Previewer* attribute), 288
- TEMPLATE (*indico.modules.attachments.preview.ImagePreviewer* attribute), 288
- TEMPLATE (*indico.modules.attachments.preview.PDFPreviewer* attribute), 288
- template (*indico.modules.designer.models.images.DesignerImageFile* attribute), 317
- template\_hook (in module *indico.core.signals.plugin*), 89
- template\_id (*indico.modules.designer.models.images.DesignerImageFile* attribute), 317
- template\_kwargs (in *indico.modules.events.logs.renderers.EventLogRendererBase* attribute), 189
- template\_kwargs (in *indico.modules.events.logs.renderers.SimpleRenderer* attribute), 189
- template\_name (in *indico.modules.events.logs.renderers.EmailRenderer* attribute), 188
- template\_name (in *indico.modules.events.logs.renderers.EventLogRendererBase* attribute), 189
- template\_name (in *indico.modules.events.logs.renderers.SimpleRenderer* attribute), 189
- TEMPLATES\_DIR (in *indico.modules.attachments.preview.Previewer* attribute), 288
- TempReservationConcurrentOccurrence (in module *indico.modules.rb.util*), 301
- TempReservationOccurrence (in module *indico.modules.rb.util*), 301
- text (*indico.modules.events.registration.models.items.RegistrationFormItem* attribute), 224
- text (*indico.modules.events.surveys.models.items.SurveyItemType* attribute), 248
- text\_color (*indico.modules.events.sessions.models.sessions.Session* attribute), 241
- text\_color (*indico.modules.events.timetable.models.breaks.Break* attribute), 254
- TextListField (class in *indico.web.forms.fields*), 336
- TextPreviewer (class in *indico.modules.attachments.preview*), 289
- theme (*indico.modules.events.models.events.Event* attribute), 127
- themes (*indico.modules.events.settings.ThemeSettingsProxy* attribute), 144, 206

ThemeSettingsProxy (class in in- *dico.modules.events.contributions.models.subcontributions.SubC*  
*dico.modules.events.settings*), 144, 206 *tribute*), 178

ticket\_on\_email (in- timetable\_entry\_created (in module in-  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *dico.core.signals.event*), 87  
*tribute*), 221 timetable\_entry\_deleted (in module in-  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *dico.core.signals.event*), 87  
*tribute*), 221 timetable\_entry\_updated (in module in-  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *dico.core.signals.event*), 87

ticket\_on\_summary\_page (in- TimetableEntry (class in in-  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *dico.modules.events.timetable.models.entries*),  
*tribute*), 221 254

ticket\_template (in- TimetableEntryType (class in in-  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *dico.modules.events.timetable.models.entries*),  
*tribute*), 221 255

ticket\_template\_id (in- timezone (indico.modules.categories.models.categories.Category  
*dico.modules.events.registration.models.forms.RegistrationForm*), 221 *tribute*), 266  
*tribute*), 221 timezone (indico.modules.events.models.events.Event  
*dico.modules.events.registration.models.registrations.Registration*), 215 *tribute*), 266

ticket\_uuid (indico.modules.events.registration.models.registrations.Registration  
*tribute*), 215 timezone (indico.web.forms.fields.IndicoDateTimeField  
*tribute*), 215 *tribute*), 340

tickets\_enabled (in- *tribute*), 340  
*dico.modules.events.registration.models.forms.RegistrationForm* (indico.web.forms.fields.OccurrencesField  
*tribute*), 221 *tribute*), 341

time (indico.modules.events.timetable.reschedule.RescheduleMode  
*tribute*), 258 *tribute*), 340  
*tribute*), 340 timezone\_field (in-  
*tribute*), 340 *tribute*), 340

TimeDeltaField (class in indico.web.forms.fields), 339 *tribute*), 340  
*tribute*), 340 timezone\_field (in-  
*tribute*), 340 *tribute*), 340

timeline (indico.modules.events.papers.models.revisions.PaperRevision  
*tribute*), 202 *tribute*), 341  
*tribute*), 202 *tribute*), 341

timeline\_item\_type (in- title (indico.modules.attachments.models.attachments.Attachment  
*dico.modules.events.models.reviews.ProposalCommentMixin*), 133 *tribute*), 282  
*tribute*), 133 title (indico.modules.attachments.models.folders.AttachmentFolder  
*dico.modules.events.models.reviews.ProposalReviewMixin*), 134 *tribute*), 285  
*tribute*), 134 title (indico.modules.categories.models.categories.Category  
*tribute*), 134 *tribute*), 266

timeline\_item\_type (in- title (indico.modules.designer.models.templates.DesignerTemplate  
*dico.modules.events.papers.models.reviews.PaperJudgmentProxy*), 199 *tribute*), 318  
*tribute*), 199 title (indico.modules.events.abstracts.models.abstracts.Abstract  
*dico.modules.events.papers.models.reviews.PaperReview*), 199 *tribute*), 147  
*tribute*), 199 *tribute*), 151

TIMELINE\_TYPE (in- *tribute*), 147  
*dico.modules.events.papers.models.reviews.PaperReview* (indico.modules.events.abstracts.models.email\_templates.AbstractE  
*tribute*), 199 *tribute*), 151

times\_changed (in module indico.core.signals.event), 87 *tribute*), 154  
*tribute*), 87 *tribute*), 154

timestamp (indico.modules.events.agreements.models.agreements.Agreement  
*tribute*), 167 *tribute*), 172  
*tribute*), 167 *tribute*), 172

timestamp (indico.modules.events.payment.models.transactions.PaymentTransaction  
*tribute*), 208 *tribute*), 173  
*tribute*), 208 *tribute*), 173

timestamp (indico.modules.rb.models.reservation\_edit\_logs.ReservationEditLog  
*tribute*), 300 *tribute*), 178  
*tribute*), 300 *tribute*), 178

timetable\_buttons (in module in- title (indico.modules.events.layout.models.menu.MenuEntry  
*dico.core.signals.event*), 87 *tribute*), 184  
*tribute*), 87 *tribute*), 184

timetable\_entries (in- title (indico.modules.events.models.events.Event *at*-  
*dico.modules.events.models.events.Event* *tribute*), 127  
*tribute*), 127 *tribute*), 127

timetable\_entry (in- title (indico.modules.events.models.persons.PersonLinkBase  
*tribute*), 130  
*tribute*), 130



[title \(indico.modules.events.models.reviews.ProposalGroupProxy attribute\), 133](#)  
[title \(indico.modules.events.papers.models.papers.Paper title\\_attr \(indico.modules.events.models.reviews.ProposalGroupProxy attribute\), 197](#)  
[title \(indico.modules.events.papers.models.review\\_question.PaperReviewQuestion attribute\), 198](#)  
[title \(indico.modules.events.registration.models.form\\_fields.RegistrationFormField attribute\), 217](#)  
[title \(indico.modules.events.registration.models.form\\_fields.RegistrationFormPersonalDataField attribute\), 219](#)  
[title \(indico.modules.events.registration.models.forms.RegistrationForm attribute\), 221](#)  
[title \(indico.modules.events.registration.models.items.PersonalDataField attribute\), 223](#)  
[title \(indico.modules.events.registration.models.items.RegistrationForm attribute\), 224](#)  
[title \(indico.modules.events.registration.models.items.RegistrationForm\(PublicDataSection attribute\), 225](#)  
[title \(indico.modules.events.registration.models.items.RegistrationForm\(Subform attribute\), 226](#)  
[title \(indico.modules.events.registration.models.items.RegistrationForm\(Ticket attribute\), 227](#)  
[title \(indico.modules.events.requests.base.RequestDefinitionBase attribute\), 239](#)  
[title \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 243](#)  
[title \(indico.modules.events.sessions.models.sessions.Session attribute\), 241](#)  
[title \(indico.modules.events.surveys.models.items.SurveyItem attribute\), 248](#)  
[title \(indico.modules.events.surveys.models.items.SurveyQuestion attribute\), 249](#)  
[title \(indico.modules.events.surveys.models.items.SurveySection attribute\), 250](#)  
[title \(indico.modules.events.surveys.models.items.SurveyText attribute\), 250](#)  
[title \(indico.modules.events.surveys.models.surveys.Survey attribute\), 247](#)  
[title \(indico.modules.events.timetable.models.breaks.Break attribute\), 254](#)  
[title \(indico.modules.events.tracks.models.tracks.Track attribute\), 260](#)  
[title \(indico.modules.events.tracks.models.tracks.Track attribute\), 260](#)  
[title \(indico.modules.news.models.news.NewsItem attribute\), 327](#)  
[title \(indico.modules.rb.models.room\\_attributes.RoomAttribute attribute\), 292](#)  
[title \(indico.modules.search.result\\_schemas.AttachmentResultSchema attribute\), 72](#)  
[title \(indico.modules.search.result\\_schemas.ContributionResultSchema attribute\), 72](#)  
[title \(indico.modules.search.result\\_schemas.EventNoteResultSchema attribute\), 73](#)  
[title \(indico.modules.search.result\\_schemas.EventResultSchema attribute\), 71](#)

[title \(indico.modules.users.models.users.PersonMixin attribute\), 272](#)  
[title \(indico.modules.attachments.models.attachments.Attachment attribute\), 133](#)  
[title \(indico.modules.attachments.models.attachments.Attachment attribute\), 133](#)  
[title \(indico.modules.events.papers.models.revisions.PaperRevisionState attribute\), 202](#)  
[title \(indico.modules.events.surveys.models.items.SurveyItem method\), 248](#)  
[title \(indico.modules.events.surveys.models.items.SurveyQuestion method\), 249](#)  
[title \(indico.modules.events.surveys.models.items.SurveySection method\), 250](#)  
[title \(indico.modules.events.surveys.models.items.SurveyText method\), 250](#)  
[title \(indico.core.oauth.models.tokens\), 309](#)  
[title \(indico.modules.rb.models.map\\_areas.MapArea attribute\), 296](#)  
[title \(indico.modules.designer.pdf\), 318](#)  
[title \(indico.modules.events.tracks.models.tracks.Track \(class in indico.modules.events.tracks.models.tracks\), 259](#)  
[title \(indico.modules.events.abstracts.models.reviews.AbstractReview attribute\), 156](#)  
[title \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 172](#)  
[title \(indico.modules.events.tracks.models.tracks.Track attribute\), 260](#)  
[title \(indico.modules.events.tracks.models.tracks.Track attribute\), 260](#)  
[title \(indico.modules.events.tracks.models.tracks.Track attribute\), 156](#)  
[title \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 172](#)  
[title \(indico.modules.events.tracks.models.principals.TrackPrincipal attribute\), 261](#)  
[title \(indico.modules.events.util\), 141](#)  
[title \(indico.modules.events.util\), 141](#)

475

`type_id(indico.modules.events.contributions.models.contributions.Contribution` (in module `in-`  
`attribute)`, 172 `dico.modules.events.abstracts.operations)`,  
`type_id(indico.modules.events.sessions.models.sessions.Session` 157  
`attribute)`, 241 `update_abstract_comment()` (in module `in-`  
`tzinfo(indico.modules.categories.models.categories.Category` `dico.modules.events.abstracts.operations)`,  
`attribute)`, 266 157  
`tzinfo(indico.modules.events.models.events.Event` `update_abstract_review()` (in module `in-`  
`attribute)`, 127 `dico.modules.events.abstracts.operations)`,  
`tzinfo(indico.web.forms.fields.IndicoDateTimeField` 157  
`attribute)`, 340 `update_break_entry()` (in module `in-`  
`dico.modules.events.timetable.operations)`,  
256  
**U**  
`under_review(indico.modules.events.abstracts.models.abstracts.AbstractPublicState` (in module `in-`  
`attribute)`, 147 `dico.modules.categories.operations)`, 268  
`undo_impersonate_user()` (in module `in-` `update_comment()` (in module `in-`  
`dico.modules.auth.util)`, 304 `dico.modules.events.papers.operations)`,  
203  
`unique_columns` (in- `update_folder_principal()` (in module `in-`  
`dico.modules.attachments.models.principals.AttachmentFolderPrincipal` `dico.modules.events.papers.operations)`,  
`attribute)`, 286 203  
`unique_columns` (in- `update_contribution()` (in module `in-`  
`dico.modules.attachments.models.principals.AttachmentPrincipal` `dico.modules.events.contributions.operations)`,  
`attribute)`, 287 179  
`unique_columns` (in- `update_data_association()` (in-  
`dico.modules.categories.models.principals.CategoryPrincipal` `dico.modules.vc.plugins.VCPluginMixin`  
`attribute)`, 267 `method)`, 316  
`unique_columns` (in- `update_data_privacy()` (in-  
`dico.modules.events.contributions.models.principals.ContributionPrincipal` `dico.modules.vc.plugins.VCPluginMixin`  
`attribute)`, 176 `method)`, 316  
`unique_columns` (in- `update_event()` (in module `in-`  
`dico.modules.events.models.principals.EventPrincipal` `dico.modules.events.operations)`, 138  
`attribute)`, 131 `update_event_label()` (in module `in-`  
`unique_columns` (in- `dico.modules.events.operations)`, 138  
`dico.modules.events.sessions.models.principals.SessionPrincipal` `update_event_protection()` (in module `in-`  
`attribute)`, 244 `dico.modules.events.operations)`, 138  
`unique_columns` (in- `update_event_type()` (in module `in-`  
`dico.modules.events.tracks.models.principals.TrackPrincipal` `dico.modules.events.operations)`, 138  
`attribute)`, 261  
`unique_columns` (in- `update_object_principals()` (in module `in-`  
`dico.modules.rb.models.blocking_principals.BlockingPrincipal` `dico.modules.events.util)`, 141  
`attribute)`, 295 `update_paper_template()` (in module `in-`  
`unique_links(indico.modules.attachments.models.folders.AttachmentFolder` `dico.modules.events.papers.operations)`, 203  
`attribute)`, 285 `update_person()` (in module `in-`  
`unique_links(indico.modules.events.notes.models.notes.EventNote` `dico.modules.events.persons.operations)`,  
`attribute)`, 192 210  
`unit_names(indico.web.forms.fields.RelativeDeltaField` `update_program()` (in module `in-`  
`attribute)`, 346 `dico.modules.events.tracks.operations)`, 261  
`unit_names(indico.web.forms.fields.TimeDeltaField` `update_reference_type()` (in module `in-`  
`attribute)`, 340 `dico.modules.events.operations)`, 139  
`unlock_event()` (in module `in-` `update_regform_item_positions()` (in mod-  
`dico.modules.events.operations)`, 138 `ule indico.modules.events.registration.util)`, 229  
`unpaid(indico.modules.events.registration.models.registrations.RegistrationState` (in module `in-`  
`attribute)`, 217 `dico.modules.events.papers.operations)`,  
204  
`unstyled(indico.modules.events.abstracts.settings.BOALinkFormat` 204  
`attribute)`, 165 `update_reviewed_for_tracks()` (in module `in-`



`dico.modules.events.abstracts.operations`), 157  
`update_reviewing_question()` (in module `indico.modules.events.operations`), 139  
`update_reviewing_roles()` (in module `indico.modules.events.papers.operations`), 204  
`update_scopes()` (in module `indico.core.oauth.models.applications.OAuthApplicationUserLink` method), 307  
`update_session()` (in module `indico.modules.events.sessions.operations`), 244  
`update_session_block()` (in module `indico.modules.events.sessions.operations`), 244  
`update_session_coordinator_privs()` (in module `indico.modules.events.sessions.operations`), 244  
`update_state()` (in module `indico.modules.events.registration.models.registration_registration` method), 215  
`update_subcontribution()` (in module `indico.modules.events.contributions.operations`), 180  
`update_team_members()` (in module `indico.modules.events.papers.operations`), 204  
`update_timetable_entry()` (in module `indico.modules.events.timetable.operations`), 256  
`update_timetable_entry_object()` (in module `indico.modules.events.timetable.operations`), 256  
`update_track()` (in module `indico.modules.events.tracks.operations`), 261  
`update_track_group()` (in module `indico.modules.events.tracks.operations`), 261  
`updated` (in module `indico.core.signals.category`), 83  
`updated` (in module `indico.core.signals.event`), 87  
`url` (`indico.modules.categories.models.categories.Category` attribute), 266  
`url` (`indico.modules.events.layout.models.menu.MenuEntryMixin` attribute), 184  
`url` (`indico.modules.events.models.events.Event` attribute), 127  
`url` (`indico.modules.events.models.references.ReferenceModelBase` attribute), 132  
`url` (`indico.modules.news.models.news.NewsItem` attribute), 327  
`url_for_login()` (in module `indico.modules.auth.util`), 304  
`url_for_logout()` (in module `indico.modules.auth.util`), 304  
`url_for_plugin()` (in module `indico.core.plugins`), 80  
`url_for_register()` (in module `indico.modules.auth.util`), 304  
`url_rule_to_angular()` (in module `indico.modules.events.registration.util`), 229  
`url_cut` (`indico.modules.events.models.events.Event` attribute), 127  
`url_template` (`indico.modules.events.models.references.ReferenceType` attribute), 132  
`url_to_static_filename()` (in module `indico.modules.events.static.util`), 263  
`urn` (`indico.modules.events.models.references.ReferenceModelBase` attribute), 132  
`use_count` (`indico.core.oauth.models.tokens.OAuthToken` attribute), 309  
`use_count` (`indico.core.oauth.models.tokens.TokenModelBase` attribute), 310  
`USE_PROXY` (built-in variable), 55  
`use_registration` (`indico.modules.users.models.users`), 272  
`user` (`indico.core.oauth.models.applications.OAuthApplicationUserLink` attribute), 307  
`user` (`indico.core.oauth.models.tokens.OAuthToken` attribute), 309  
`user` (`indico.modules.attachments.models.attachments.Attachment` attribute), 282  
`user` (`indico.modules.attachments.models.attachments.AttachmentFile` attribute), 283  
`user` (`indico.modules.attachments.models.principals.AttachmentFolderPrincipal` attribute), 286  
`user` (`indico.modules.attachments.models.principals.AttachmentPrincipal` attribute), 287  
`user` (`indico.modules.categories.models.principals.CategoryPrincipal` attribute), 267  
`user` (`indico.modules.events.abstracts.models.comments.AbstractComment` attribute), 149  
`user` (`indico.modules.events.abstracts.models.email_logs.AbstractEmailLog` attribute), 150  
`user` (`indico.modules.events.abstracts.models.reviews.AbstractReview` attribute), 156  
`user` (`indico.modules.events.agreements.models.agreements.Agreement` attribute), 167  
`user` (`indico.modules.events.contributions.models.principals.ContributionPrincipal` attribute), 176  
`user` (`indico.modules.events.logs.models.entries.EventLogEntry` attribute), 187  
`user` (`indico.modules.events.models.persons.EventPerson` attribute), 129  
`user` (`indico.modules.events.models.principals.EventPrincipal` attribute), 131  
`user` (`indico.modules.events.models.settings.EventSettingPrincipal` attribute), 136  
`user` (`indico.modules.events.notes.models.notes.EventNoteRevision` attribute), 192

user_id(indico.modules.events.papers.models.comments.PaperReviewComment), 156	user_id(indico.modules.events.agreements.models.agreements.Agreement), 156
attribute), 194	attribute), 167
user_id(indico.modules.events.papers.models.competences.PaperCompetence), 195	user_id(indico.modules.events.contributions.models.principals.ContributionPrincipal), 176
user_id(indico.modules.events.papers.models.reviews.PaperReviewComment), 200	user_id(indico.modules.events.logs.models.entries.EventLogEntry), 187
user_id(indico.modules.events.registration.models.registrations.Registration), 215	user_id(indico.modules.events.models.persons.EventPerson), 129
user_id(indico.modules.events.sessions.models.principals.SessionPrincipal), 244	user_id(indico.modules.events.models.principals.EventPrincipal), 131
user_id(indico.modules.events.surveys.models.submissions.SurveySubmission), 251	user_id(indico.modules.events.models.settings.EventSettingPrincipal), 136
user_id(indico.modules.events.tracks.models.principals.TrackPrincipal), 261	user_id(indico.modules.events.notes.models.notes.EventNoteRevision), 192
user_id(indico.modules.rb.models.blocking_principals.BlockingPrincipal), 295	user_id(indico.modules.events.papers.models.comments.PaperReviewComment), 194
user_id(indico.modules.search.result_schemas.AttachmentResultSchema), 72	user_id(indico.modules.events.papers.models.competences.PaperCompetence), 195
user_id(indico.modules.search.result_schemas.EventNoteResultSchema), 73	user_id(indico.modules.events.papers.models.reviews.PaperReviewComment), 200
user_id(indico.modules.users.models.settings.UserSetting), 277	user_id(indico.modules.events.registration.models.registrations.Registration), 215
user_backref_name (in-dico.modules.events.abstracts.models.comments.AbstractComment), 149	user_id(indico.modules.events.sessions.models.principals.SessionPrincipal), 244
user_backref_name (in-dico.modules.events.papers.models.comments.PaperReviewComment), 194	user_id(indico.modules.events.surveys.models.submissions.SurveySubmission), 251
user_competences (in-dico.modules.events.papers.models.call_for_papers.CallForPapers), 194	user_id(indico.modules.events.tracks.models.principals.TrackPrincipal), 261
user_data(indico.modules.auth.models.registration_requests.RegistrationRequest), 303	user_id(indico.modules.rb.models.blocking_principals.BlockingPrincipal), 295
user_data(indico.modules.events.registration.models.registrations.RegistrationData), 216	user_id(indico.modules.users.models.affiliations.UserAffiliation), 276
user_id(indico.core.oauth.models.applications.OAuthApplication), 307	user_id(indico.modules.users.models.emails.UserEmail), 276
user_id(indico.modules.attachments.models.attachments.Attachment), 282	user_id(indico.modules.users.models.settings.UserSetting), 277
user_id(indico.modules.attachments.models.attachments.AttachmentFile), 283	user_id(indico.modules.users.models.suggestions.SuggestedCategory), 277
user_id(indico.modules.attachments.models.principals.AttachmentPrincipal), 286	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.attachments.models.principals.AttachmentPrincipal), 287	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.auth.models.identities.Identity), 303	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.categories.models.principals.CategoryPrincipal), 267	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.events.abstracts.models.comments.AbstractComment), 149	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry), 150	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185
user_id(indico.modules.events.abstracts.models.reviews.AbstractReview), 147	user_id(indico.modules.events.layout.models.menu.MenuEntryType), 185

tribute), 79

user\_settings\_converters (in-  
dico.core.plugins.IndicoPlugin  
79 attribute), 79

UserAffiliation (class in in-  
dico.modules.users.models.affiliations), 276

UserEmail (class in in-  
dico.modules.users.models.emails), 276

users (indico.modules.events.abstracts.models.reviews.AbstractCommentVisibility), 155

users (indico.modules.events.papers.models.reviews.PaperCommentVisibility), 199

UserSetting (class in in-  
dico.modules.users.models.settings), 277

UserSettingsProxy (class in in-  
dico.modules.users.models.settings), 277

UserName (class in in-  
dico.modules.users.models.users), 275

uuid (indico.modules.events.abstracts.models.abstracts.AbstractRoomEventAssociation), 147

uuid (indico.modules.events.agreements.models.agreements.AgreementLinkType), 167

uuid (indico.modules.events.models.static\_list\_links.StaticListLink), 137

uuid (indico.modules.events.registration.models.invitations.RegistrationInvitation), 222

uuid (indico.modules.events.registration.models.registrations.Registration), 215

uuid (indico.modules.events.surveys.models.surveys.Survey), 247

value (indico.modules.rb.models.room\_attributes.RoomAttributeAssociation), 293

value (indico.modules.users.models.settings.UserSetting attribute), 277

vc\_room (indico.modules.vc.models.vc\_rooms.VCRoomEventAssociation attribute), 313

vc\_room\_attach\_form (in-  
dico.modules.vc.plugins.VCPluginMixin attribute), 316

vc\_room\_form (indico.modules.vc.plugins.VCPluginMixin attribute), 316

vc\_room\_id (indico.modules.vc.models.vc\_rooms.VCRoomEventAssociation attribute), 313

VCPluginMixin (class in indico.modules.vc.plugins), 314

VCRoom (class in indico.modules.vc.models.vc\_rooms), 312

VCRoomError, 316

VCRoomEventAssociation (class in in-  
dico.modules.vc.models.vc\_rooms), 312

Venue (class in in-  
dico.modules.vc.models.vc\_rooms), 313

VCRoomStatus (class in in-  
dico.modules.vc.models.vc\_rooms), 313

venue\_name (indico.modules.events.contributions.models.subcontributions.SubContribution), 178

venue\_name (indico.modules.search.result\_schemas.LocationResultSchema attribute), 73

verbose\_name (indico.modules.rb.models.rooms.Room attribute), 292

verbose\_title (in-  
dico.modules.events.abstracts.models.abstracts.AbstractRoomEventAssociation attribute), 147

valid (indico.modules.rb.models.reservation\_occurrences.ReservationOccurrenceStatus attribute), 301

valid\_currencies (in-  
dico.modules.events.payment.plugins.PaymentPluginMixin attribute), 210

value (indico.modules.categories.models.settings.CategorySetting attribute), 268

value (indico.modules.events.abstracts.models.review\_ratings.AbstractReviewRating attribute), 154

value (indico.modules.events.contributions.models.references.ContributionReference attribute), 177

value (indico.modules.events.contributions.models.references.SubContributionReference attribute), 177

value (indico.modules.events.models.references.EventReference attribute), 131

value (indico.modules.events.models.references.ReferenceModelBase attribute), 132

value (indico.modules.events.models.settings.EventSetting attribute), 135

value (indico.modules.events.papers.models.review\_ratings.PaperReviewRating attribute), 198

version\_of (indico.modules.attachments.models.attachments.Attachment), 283

version\_of (indico.modules.designer.models.images.DesignerImageFile attribute), 217

version\_of (indico.modules.events.layout.models.images.ImageFile attribute), 183

versioned\_data (in-  
dico.modules.events.registration.models.form\_fields.RegistrationFormField attribute), 218

versioned\_data (in-  
dico.modules.events.registration.models.form\_fields.RegistrationFormField attribute), 218

view\_data (indico.modules.events.registration.models.form\_fields.RegistrationFormField attribute), 218

[view\\_data \(indico.modules.events.registration.models.form\\_fields.RegistrationFormFields.TrackRoleField attribute\), 219](#)  
[view\\_data \(indico.modules.events.registration.models.items.RegistrationFormFields.ContributionPersonLink attribute\), 224](#)  
[view\\_data \(indico.modules.events.registration.models.items.RegistrationFormFields.SubContributionPersonLink attribute\), 225](#)  
[view\\_data \(indico.modules.events.registration.models.items.RegistrationFormFields.EventPersonLinkListField attribute\), 226](#)  
[view\\_data \(indico.modules.events.registration.models.items.RegistrationFormFields.PersonLinkListFieldBase attribute\), 227](#)  
[visibility \(indico.modules.categories.models.categories.CategoryField attribute\), 266](#)  
[visibility \(indico.modules.events.abstracts.models.comments.AbstractCommentField attribute\), 149](#)  
[visibility \(indico.modules.events.abstracts.models.reviews.AbstractReviewField attribute\), 156](#)  
[visibility \(indico.modules.events.contributions.models.files.ContributionFileField attribute\), 173](#)  
[visibility \(indico.modules.events.models.events.Event widget \(indico.web.forms.fields.FileField attribute\), attribute\), 127](#)  
[visibility \(indico.modules.events.papers.models.comments.PaperReviewField widget \(indico.web.forms.fields.HiddenFieldList attribute\), 195](#)  
[visibility \(indico.modules.events.papers.models.reviews.PaperReview widget \(indico.web.forms.fields.IndicoDateField attribute\), 200](#)  
[visibility\\_horizon\\_query \(indico.modules.categories.models.categories.Category widget \(indico.web.forms.fields.IndicoDateTimeField attribute\), 266](#)  
[visible \(\) \(indico.modules.events.layout.util.MenuEntryData widget \(indico.web.forms.fields.IndicoEmailRecipientsField attribute\), 186](#)  
[visible\\_categories\\_query \(indico.modules.categories.models.categories.Category widget \(indico.web.forms.fields.IndicoEnumRadioField attribute\), 266](#)  
[visible\\_categories\\_query \(indico.modules.categories.models.categories.Category widget \(indico.web.forms.fields.IndicoEnumSelectField attribute\), 266](#)  
[visible\\_categories\\_query \(indico.modules.categories.models.categories.Category widget \(indico.web.forms.fields.IndicoLocationField attribute\), 266](#)  
[warning \(indico.modules.categories.models.categories.EventMessageField widget \(indico.web.forms.fields.IndicoMarkdownField attribute\), 267](#)  
[was\\_survey\\_submitted \(\) \(in module indico.modules.events.surveys.util\), 253](#)  
[WEEK \(indico.modules.rb.models.reservations.RepeatFrequency widget \(indico.web.forms.fields.IndicoPasswordField attribute\), 296](#)  
[week\\_day\\_data \(indico.web.forms.fields.IndicoWeekDayRepetitionField widget \(indico.web.forms.fields.IndicoProtectionField attribute\), 347](#)  
[WEEK\\_DAY\\_NUMBER\\_CHOICES \(indico.web.forms.fields.IndicoWeekDayRepetitionField widget \(indico.web.forms.fields.IndicoQuerySelectMultipleCheckboxField attribute\), 346](#)  
[widget \(indico.modules.categories.fields.CategoryField widget \(indico.web.forms.fields.IndicoRadioField attribute\), 333](#)  
[widget \(indico.modules.events.abstracts.fields.AbstractField widget \(indico.web.forms.fields.IndicoSelectMultipleCheckboxField attribute\), 328](#)  
[widget \(indico.modules.events.abstracts.fields.AbstractPersonLinkField widget \(indico.web.forms.fields.IndicoStaticTextField attribute\), 329](#)  
[widget \(indico.modules.events.abstracts.fields.EmailRuleListField widget \(indico.web.forms.fields.IndicoTagListField attribute\), 330](#)  
[widget \(indico.modules.events.abstracts.fields.EmailRuleListField widget \(indico.web.forms.fields.IndicoTimeField attribute\), 330](#)

widget (*indico.web.forms.fields.IndicoWeekDayRepetitionField* attribute), 347 189  
 widget (*indico.web.forms.fields.MultipleItemsField* attribute), 343 WPJinjaMixinPlugin (class in *indico.core.plugins*), 80  
 widget (*indico.web.forms.fields.MultiStringField* attribute), 342 X  
 widget (*indico.web.forms.fields.OccurrencesField* attribute), 341 XELATEX\_PATH (built-in variable), 54  
 widget (*indico.web.forms.fields.OverrideMultipleItemsField* attribute), 343 Z  
 widget (*indico.web.forms.fields.PrincipalField* attribute), 344 ZipGeneratorMixin (class in *indico.modules.events.util*), 139  
 widget (*indico.web.forms.fields.PrincipalListField* attribute), 344  
 widget (*indico.web.forms.fields.RelativeDeltaField* attribute), 346  
 widget (*indico.web.forms.fields.TimeDeltaField* attribute), 340  
 width (*indico.modules.designer.pdf.TplData* attribute), 319  
 width\_cm (*indico.modules.designer.pdf.TplData* attribute), 319  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholder* attribute), 320  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderB* attribute), 320  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderC* attribute), 321  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderD* attribute), 321  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNamePlaceholder* attribute), 320  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNamePlaceholderB* attribute), 320  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNamePlaceholderC* attribute), 321  
 with\_title (*indico.modules.designer.placeholders.RegistrationFullNamePlaceholderD* attribute), 321  
 withdraw () (*indico.modules.events.requests.base.RequestDefinitionBase* class method), 239  
 withdraw\_abstract () (in module *indico.modules.events.abstracts.operations*), 157  
 withdrawn (*indico.modules.events.abstracts.models.abstracts.AbstractPublicState* attribute), 147  
 withdrawn (*indico.modules.events.abstracts.models.abstracts.AbstractState* attribute), 148  
 withdrawn (*indico.modules.events.registration.models.registrations.RegistrationState* attribute), 217  
 withdrawn (*indico.modules.events.requests.models.requests.RequestState* attribute), 237  
 WORKER\_NAME (built-in variable), 57  
 WPEventManagement (class in *indico.modules.events.management.views*),