Python flask.request.mimetype() Examples

The following are code examples for showing how to use *flask.request.mimetype()*. They are from open source Python projects. You can vote up the examples you like or vote down the ones you don't like.

Example 1

Project: *git-webhook* Author: *NetEaseGame* File: *validator.py* MIT License 6 vo

```
def get_data(self):
        """
        Get request data based on request.method and request.mimetype

        Returns:
            A regular dict which can be modified(scheme will modify data
            on validating)
        """
        if request.method in ['GET', 'DELETE']:
            return request.args.to_dict()
        else:
            if request.mimetype == 'application/json':
                data = request.get_json()
                if not isinstance(data, collections.Mapping):
                    self.handle_error('JSON content must be object')
                return data
            else:
                return request.form.to_dict()
```

Example 2

Project: *woodbox* Author: *patrickfournier* File: *record_api.py* Apache License 2.0 6 vo

```
def post(self):
        if request.mimetype != 'application/vnd.api+json':
            return '', 415, {'Accept-Patch': 'application/vnd.api+json'}

        input_data = request.get_json(force=True) or {}
        schema = self.schema_class()

        try:
            data, _ = schema.load(input_data)
        except ValidationError as err:
            abort(415, errors=[err.args[0]])
        except Exception as err:
            abort(422, errors=[err.args[0]])

        new_item = self.model_class(**data)
        db.session.add(new_item)
        db.session.commit()

        return (self.schema_class().dump(new_item, many=False).data,
                200, {'Content-Location': url_for(self.record_api.scoped_endpoint(
```

Example 3

Project: *trojandroid_server* Author: *remijouannet* File: *app.py* GNU General Public License v3.0 6 vo

```
def action(self):
        sha1 = hashlib.sha1()
        sha1.update(KEY)

        try:
            SHA = request.headers.get('Authorization').split('::::')[1]
            MAC = request.headers.get('Authorization').split('::::')[0]
            IP = request.remote_addr
            if SHA != sha1.hexdigest():
                return Response(self.null, status=401)
        except Exception:
            return Response(self.null, status=401)


        for arg, value in sorted(vars(self.args).items()):
            if value and not self.nullAction and arg not in self.excludeArgs:
                return Response(json.dumps({arg: value}), status=200, mimetype='a
        return self.null
```

Example 4

Project: *har-sanitizer* Author: *google* File: *decorators.py* Apache License 2.0 5 vo

```
def accept(mimetype):
    def decorator(func):
        """
        Decorator which returns a 406 Not Acceptable if the client won't accept
        a certain mimetype
        """
        @wraps(func)
        def wrapper(*args, **kwargs):
            if mimetype in request.accept_mimetypes:
                return func(*args, **kwargs)
            message = "Request must accept {} data".format(mimetype)
            data = json.dumps({"message": message})
            return Response(data, 406, mimetype="application/json")
        return wrapper
    return decorator
```

Example 5

Project: *har-sanitizer* Author: *google* File: *decorators.py* Apache License 2.0 5 vo

```
def require(mimetype):
    def decorator(func):
        """
        Decorator which returns a 415 Unsupported Media Type if the client sends
        something other than a certain mimetype
        """
        @wraps(func)
        def wrapper(*args, **kwargs):
            if (request.mimetype ==  mimetype):
                return func(*args, **kwargs)
            message = "Request must contain {} data".format(mimetype)
            data = json.dumps({"message": message})
            return Response(data, 415, mimetype="application/json")
        return wrapper
    return decorator
```

Example 6

Project: *graphene-file-upload* Author: *lmcgartland* File: *__init__.py* MIT License 5 vo

```
def parse_body(self):
        """Handle multipart request spec for multipart/form-data"""
        content_type = request.mimetype
        if content_type == 'multipart/form-data':
            operations = load_json_body(request.form.get('operations', '{}'))
            files_map = load_json_body(request.form.get('map', '{}'))
            return place_files_in_operations(
                operations,
                files_map,
                request.files
            )
        return super(FileUploadGraphQLView, self).parse_body()
```

Example 7

Project: *invenio-records-rest* Author: *inveniosoftware* File: *errors.py* MIT License 5 vo

```
def __init__(self, content_type=None, **kwargs):
        """Initialize exception."""
        super(RESTException, self).__init__(**kwargs)
        content_type = content_type or request.mimetype
        self.description = 'Unsupported media type "{0}".'.format(content_type)
```

Example 8

Project: *flump* Author: *rolepoint* File: *web_utils.py* MIT License 5 vo

```
def get_json():
    """
    Returns the request.json if we have the correct MIMETYPE.
    """
    if request.mimetype and request.mimetype not in ALLOWED_MIMETYPES:
        raise UnsupportedMediaType

    return request.get_json(force=True)
```

Example 9

Project: *cauldron* Author: *sernst* File: *execution.py* MIT License 5 vo

```
def parse_command_args(response: 'Response') -> typing.Tuple[str, str]:
    """

    :param response:
        The response object to modify with status or error data
    :return:
        A tuple where the first element if the name of the command
        to execute, and the second is a string representing the arguments
        to apply to that command.
    """

    cmd = None
    parts = None
    name = None
    args = None
    request_args = arguments.from_request()

    try:
```

```
        cmd = request_args.get('command', '')
        parts = [x.strip() for x in cmd.split(' ', 1)]
        name = parts[0].lower()

        args = request_args.get('args', '')
        if not isinstance(args, str):
            args = ' '.join(args)
        args += ' {}'.format(parts[1] if len(parts) > 1 else '').strip()
    except Exception as err:
        response.fail(
            code='INVALID_COMMAND',
            message='Unable to parse command',
            cmd=cmd if cmd else '',
            parts=parts,
            name=name,
            args=args,
            error=err,
            mime_type='{}'.format(request.mimetype),
            request_data='{}'.format(request.data),
            request_args=request_args
        )

    return name, args
```

Example 10

Project: *woodbox* Author: *patrickfournier* File: *record_api.py* Apache License 2.0 5 vo

```
def patch(self, item_id):
        if request.mimetype != 'application/vnd.api+json':
            return '', 415, {'Accept-Patch': 'application/vnd.api+json'}

        input_data = request.get_json(force=True) or {}
        schema = self.schema_class()

        try:
            data, _ = schema.load(input_data, partial=True)
        except ValidationError as err:
            abort(415, errors=[err.args[0]])
        except Exception as err:
            abort(422, errors=[err.args[0]])

        item, exists = self._get_item(item_id, 'update', check_existence=True)

        if not item:
            # According to RFC5789, we may create the ressource, but we do not.
            if exists is None:
                abort(500)
            elif not exists:
                abort(404)
            else:
                abort(403)
        else:
            for key in data:
                setattr(item, key, data[key])

            msg = item.checkUpdatePrecondition()
            if msg:
                abort(400, errors=[msg])
            else:
                db.session.commit()
                return '', 204, {'Content-Location': url_for(self.scoped_endpoint(
```

Example 11

Project: *flask-autofixture* Author: *janukobytsch* File: *fixture.py* MIT License 5 vo

```
def from_request(cls, request, app, name):
        if not Fixture.is_supported(request.mimetype):
            raise TypeError
        fixture = cls(request.data, name, app.name, request.path,
                      request.method, is_response=False)
        return fixture
```

Example 12

Project: *flask-autofixture* Author: *janukobytsch* File: *fixture.py* MIT License 5 vo

```
def from_response(cls, response, app, name):
        if not Fixture.is_supported(response.mimetype):
            raise TypeError
        fixture = cls(response.get_data(), name, app.name, request.path,
                      request.method)
        return fixture
```

Example 13

Project: *trojandroid_server* Author: *remijouannet* File: *app.py* GNU General Public License v3.0 5 vo

```
def result(self):
        sha1 = hashlib.sha1()
        sha1.update(KEY)

        SHA = request.headers.get('Authorization').split('::::')[1]
        MAC = request.headers.get('Authorization').split('::::')[0]
        IP = request.remote_addr

        if SHA == sha1.hexdigest():
            print(IP + " " + MAC)
            if request.mimetype == "application/json":
                try:
                    resultjson = json.dumps(request.get_json(), indent=3, sort_key
                    print(resultjson)
                except Exception:
                    print(str(request.data))
            elif request.mimetype == "multipart/form-data":
                fileresult = expanduser("~") + "/result"
                print(fileresult)
                request.files['filedata'].save(fileresult)
            else:
                print(str(request.data))

            self.nullAction = True
            self.stop()
            return Response(self.null, status=200)
        else:
            print(request.remote_addr + "Wrong KEY")
            return Response(self.null, status=401)
```

Example 14

Project: *docker* Author: *getavalon* File: *flask.py* MIT License 5 vo

```
def get_http_info(self, request):
        """
        Determine how to retrieve actual data by using request.mimetype.
        """
        if self.is_json_type(request.mimetype):
            retriever = self.get_json_data
        else:
            retriever = self.get_form_data
        return self.get_http_info_with_retriever(request, retriever)
```

Example 15

Project: *wazo-dird* Author: *wazo-platform* File: *http.py* GNU General Public License v3.0 5 vo

```
def get(self):
        user_uuid = _get_calling_user_uuid()

        contacts = self.personal_service.list_contacts_raw(user_uuid)

        mimetype = request.mimetype
        if not mimetype:
            args = parser.parse_args()
            mimetype = args.get('format', None)

        return self.contacts_formatter(mimetype)(contacts)
```

Example 16

Project: *wazo-dird* Author: *wazo-platform* File: *http.py* GNU General Public License v3.0 5 vo

```
def contacts_formatter(cls, mimetype):
        formatters = {'text/csv': cls.format_csv, 'application/json': cls.format_j
        return formatters.get(mimetype, cls.format_json)
```

Example 17

Project: *invenio-rest* Author: *inveniosoftware* File: *decorators.py* MIT License 5 vo

```
def require_content_types(*allowed_content_types):
    r"""Decorator to test if proper Content-Type is provided.

    :param \*allowed_content_types: List of allowed content types.
    :raises invenio_rest.errors.InvalidContentType: It's rised if a content
        type not allowed is required.
    """
    def decorator(f):
        @wraps(f)
        def inner(*args, **kwargs):
            if request.mimetype not in allowed_content_types:
                raise InvalidContentType(allowed_content_types)
            return f(*args, **kwargs)
        return inner
    return decorator
```

Example 18

Project: *serverless-ping* Author: *nickromano* File: *flask.py* MIT License 5 vo

```
def get_http_info(self, request):
        """
```

```
        Determine how to retrieve actual data by using request.mimetype.
        """
        if self.is_json_type(request.mimetype):
            retriever = self.get_json_data
        else:
            retriever = self.get_form_data
        return self.get_http_info_with_retriever(request, retriever)
```