Python flask.request.content_type() Examples

The following are code examples for showing how to use *flask.request.content_type()*. 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: *ssrspeed_backup* Author: *mazhenting* File: *getpostdata.py* GNU General Public License v3.0 7 vo

```
def getPostData():
        #print(request.content_type)
        data = {}
        if (request.content_type.startswith('application/json')):
                data = request.get_data()
                return json.loads(data.decode("utf-8"))
        elif(request.content_type.startswith("application/x-www-form-urlencoded")
                #print(1)
                #print(urllib.parse.parse_qs(request.get_data().decode("utf-8")))
                return parse_qs_plus(urllib.parse.parse_qs(request.get_data().decc
        else:
                for key, value in request.form.items():
                        if key.endswith('[]'):
                                data[key[:-2]] = request.form.getlist(key)
                        else:
                                data[key] = value
                return data
```

Example 2

Project: *flask-jwt-extended* Author: *vimalloc* File: *view_decorators.py* MIT License 6 vo

```
def _decode_jwt_from_json(request_type):
    if request.content_type != 'application/json':
        raise NoAuthorizationError('Invalid content-type. Must be application/jsor

    if request_type == 'access':
        token_key = config.json_key
    else:
        token_key = config.refresh_json_key

    try:
        encoded_token = request.json.get(token_key, None)
        if not encoded_token:
            raise BadRequest()
    except BadRequest:
        raise NoAuthorizationError('Missing "{}" key in json data.'.format(token_k

    return encoded_token, None
```

Example 3

Project: *libresign* Author: *this-is-ari* File: *main.py* MIT License 6 vo

```
def locate_fields():
    if request.content_type != 'application/pdf':
        return json.dumps({
            'msg': 'Request did not have content type "application/pdf"'
        }), 415

    try:
        return json.dumps(
            parse_pdf(BytesIO(request.stream.read())),
            iterable_as_array=True
        ), 200

    except PDFSyntaxError as e:
        return json.dumps({
            'msg': 'Invalid PDF',
            'err': str(e)
        }), 400
```

Example 4

Project: *AIforEarth-API-Development* Author: *microsoft* File: *ai4e_service.py* MIT License 6 vo

```
def before_request(self):
        # Don't accept a request if SIGTERM has been called on this instance.
        if (self.is_terminating):
            print('Process is being terminated. Request has been denied.')
            abort(503, {'message': 'Service is busy, please try again later.'})

        if request.path in self.func_properties:
            if (self.func_request_counts[request.path] + 1 > self.func_properties[
                print('Service is busy. Request has been denied.')
                abort(503, {'message': 'Service is busy, please try again later.'}

            if (self.func_properties[request.path][CONTENT_TYPE_KEY_NAME] and not
                print('Invalid content type. Request has been denied.')
                abort(401, {'message': 'Content-type must be ' + self.func_propert

            if (self.func_properties[request.path][CONTENT_MAX_KEY_NAME] and reque
                print('Request is too large. Request has been denied.')
                abort(413, {'message': 'Request content too large (' + str(request
```

Example 5

Project: *SSRSpeed* Author: *NyanChanMeow* File: *getpostdata.py* GNU General Public License v3.0 6 vo

```
def getPostData():
        #print(request.content_type)
        data = {}
        if (request.content_type.startswith('application/json')):
                data = request.get_data()
                return json.loads(data.decode("utf-8"))
        elif(request.content_type.startswith("application/x-www-form-urlencoded")
                #print(1)
                #print(urllib.parse.parse_qs(request.get_data().decode("utf-8")))
                return parse_qs_plus(urllib.parse.parse_qs(request.get_data().deco
        else:
                for key, value in request.form.items():
                        if key.endswith('[]'):
                                data[key[:-2]] = request.form.getlist(key)
                        else:
                                data[key] = value
                return data
```

Example 6

Project: *pm-odsc-restful* Author: *mlpiper* File: *main.py* Apache License 2.0 6 vo

```
def output_json(data, code, headers=None):
    """
    This method is used to serialize the python dict to a json
    :param data:
    :param code:
    :param headers:
    :return:
    """
    content_type = 'application/json'
    dumped = json.dumps(data)
    if headers:
        headers.update({'Content-Type': content_type})
    else:
        headers = {'Content-Type': content_type}
    response = make_response(dumped, code, headers)
    return response
```

Example 7

Project: *CTFd* Author: *CTFd* File: *__init__.py* Apache License 2.0 6 vo

```
def require_verified_emails(f):
    """
    Decorator to restrict an endpoint to users with confirmed active email address
    :param f:
    :return:
    """

    @functools.wraps(f)
    def _require_verified_emails(*args, **kwargs):
        if get_config("verify_emails"):
            if current_user.authed():
                if (
                    current_user.is_admin() is False
                    and current_user.is_verified() is False
                ):  # User is not confirmed
                    if request.content_type == "application/json":
                        abort(403)
                    else:
                        return redirect(url_for("auth.confirm"))
        return f(*args, **kwargs)

    return _require_verified_emails
```

Example 8

Project: *CTFd* Author: *CTFd* File: *__init__.py* Apache License 2.0 6 vo

```
def authed_only(f):
    """
    Decorator that requires the user to be authenticated
    :param f:
    :return:
    """

    @functools.wraps(f)
    def authed_only_wrapper(*args, **kwargs):
        if authed():
            return f(*args, **kwargs)
        else:
            if request.content_type == "application/json" or request.accept_mimet
                abort(403)
            else:
                return redirect(url_for("auth.login", next=request.full_path))

    return authed_only_wrapper
```

Example 9

Project: *CTFd* Author: *CTFd* File: *__init__.py* Apache License 2.0 6 vo

```
def admins_only(f):
    """
    Decorator that requires the user to be authenticated and an admin
    :param f:
    :return:
    """

    @functools.wraps(f)
    def admins_only_wrapper(*args, **kwargs):
        if is_admin():
            return f(*args, **kwargs)
        else:
            if request.content_type == "application/json":
                abort(403)
            else:
                return redirect(url_for("auth.login", next=request.full_path))

    return admins_only_wrapper
```

Example 10

Project: *CTFd* Author: *CTFd* File: *visibility.py* Apache License 2.0 6 vo

```
def check_account_visibility(f):
    @functools.wraps(f)
    def _check_account_visibility(*args, **kwargs):
        v = get_config("account_visibility")
        if v == "public":
            return f(*args, **kwargs)

        elif v == "private":
            if authed():
                return f(*args, **kwargs)
            else:
                if request.content_type == "application/json":
                    abort(403)
                else:
                    return redirect(url_for("auth.login", next=request.full_path))

        elif v == "admins":
            if is_admin():
                return f(*args, **kwargs)
            else:
                abort(404)

    return _check_account_visibility
```

Example 11

Project: *flask-graphql-rest* Author: *biosustain* File: *__init__.py* MIT License 5 vo

```
def get_variable_values(self):
        if request.method == 'GET':
            return request.args
        elif request.method == 'POST':
            if request.content_type == 'application/json':
                return request.json
            else:
                return request.data

        raise NotImplementedError
```

Example 12

Project: *covador* Author: *baverman* File: *flask.py* MIT License 5 vo

```
def get_form():
    try:
        return request._covador_form
    except AttributeError:
        ctype = request.content_type or ''
        if ctype.startswith('multipart/form-data'):
            form = request.form.to_dict(False)
        elif ctype.startswith('application/x-www-form-urlencoded'):
            form = parse_qs(request.get_data(parse_form_data=False))
        else:
            form = {}
        request._covador_form = form
        return form
```

Example 13

Project: *covador* Author: *baverman* File: *flask.py* MIT License 5 vo

```
def get_json():
    ctype = request.content_type or ''
    if ctype.startswith('application/json'):
        return request.get_json()
    return {}
```

Example 14

Project: *pia* Author: *soasme* File: *view.py* ISC License 5 vo

```
def builtin_jq():
    """
    Builtin program: `jq`.
    It will run a `jq` progress and return a json object.
    """
    program = request.args.get('program', ".")
    command = request.data
    try:
        data = jq(program, command)
        resp = make_response(data)
        resp.content_type = 'application/json'
        return resp
    except InvalidJQFilter as exception:
        return jsonify(message=str(exception)), 400
```

Example 15

Project: *pia* Author: *soasme* File: *view.py* ISC License 5 vo

```
def builtin_echo():
    """
    Builtin program: `echo`.
    It will response form data.
    """
    resp = make_response(request.data)
    resp.content_type = request.content_type
    return resp
```

Example 16

Project: *heroku-python-boilerplate* Author: *chavli* File: *decorators.py* GNU General Public License v3.0 5 vo

```
def validate_jpeg_binary(func):
    """ checks the mimetype and the binary data to ensure it's a JPEG """
    @wraps(func)
    def wrapper(*args, **kwargs):
        if request.content_type != "image/jpeg":
            return ErrorResponseJson("invalid content type: {}".format(request.co
        if imghdr.test_jpeg(request.data, None) != "jpeg":
            return ErrorResponseJson("invalid jpeg data").make_response()
        return func(*args, **kwargs)
    return wrapper
```

Example 17

Project: *intracing* Author: *inspectorioinc* File: *flask.py* MIT License 5 vo

```
def enter_request_context(cls):
        span = opentracing.tracer.get_span()
        cls.set_request_tags(
            span,
            request.method,
            request.url,
            request.user_agent.string,
            request.content_type,
            request.data,
        )
        request.tracing_context = RequestContextManager(span)
        request.tracing_context.__enter__()
```

Example 18

Project: *intracing* Author: *inspectorioinc* File: *flask.py* MIT License 5 vo

```
def exit_request_context(cls, response):
        span = opentracing.tracer.get_span()
        body = response.data if not response.direct_passthrough else None
        cls.set_response_tags(
            span,
            response.status_code,
            response.content_type,
            body,
        )
        request.tracing_context.__exit__()
        return response
```

Example 19

Project: *jenkins-x-seldon-core-sandbox*   Author: *SeldonIO*   File: *flask_utils.py*   Apache License 2.0   5 vo

```
def get_request() -> Dict:
    """
    Parse a request to get JSON dict

    Returns
    -------
       JSON Dict

    """

    if (
        request.content_type is not None
        and "multipart/form-data" in request.content_type
    ):
        return get_multi_form_data_request()

    j_str = request.form.get("json")
    if j_str:
        message = json.loads(j_str)
    else:
        j_str = request.args.get("json")
        if j_str:
            message = json.loads(j_str)
        else:
            message = request.get_json()
            if message is None:
                raise SeldonMicroserviceException("Can't find JSON in data")
    if message is None:
        raise SeldonMicroserviceException("Invalid Data Format - empty JSON")
    return message
```

Example 20

Project: *Schurz*   Author: *Yensan*   File: *drops.py*   BSD 3-Clause "New" or "Revised" License   5 vo

```
def check_ContentType():
    """
    HowTo make Pre-Processing for all requests.
    But: it also can be managed in View Class, just like django-rest-framework
    """
    if request.method != 'GET':
        if (not request.content_type) or ('application/json' not in request.conte
            msg = jsonify(
                {"error": "content_type:'%s' not supported, please use 'applicati
            )
            return msg, 400
```

Example 21

Project: *flask-sqla2api*   Author: *acifani*   File: *models.py*   BSD 3-Clause "New" or "Revised" License   5 vo

```
def get_data(self, request):
        if "multipart/form-data" in request.content_type:
            return request.form.to_dict()
        if request.content_type == "application/json":
            return request.get_json()
        return
```

Example 22

Project: *pm-odsc-restful* Author: *mlpiper* File: *main.py* Apache License 2.0 5 vc

```
def post(self):
        #print("Got predict")
        #print(request.method)
        model = current_app.config["model"]
        if model is None:
            return {'error': "model is not loaded"}
        #print(request)
        #print(request.content_type)
        content = request.get_json(force=True)
        if content is None:
            return {'error': 'content of request is None'}, 404

        # print("Content is: {}".format(content))
        if "sample" not in content:
            return {'error': "sample key is not found in content"}

        sample = content["sample"]

        start = time.time()
        np_sample = np.asarray(sample).reshape(-1, len(sample))
        #print("Sample: {}".format(np_sample))

        np_pred = model.predict(np_sample)
        np_pred_prob = model.predict_proba(np_sample)[0]

        list_pred = list(np_pred)
        list_pred_prob = list(np_pred_prob)

        list_pred = list(map(float, list_pred))
        list_pred_prob = list(map(float, list_pred_prob))
        end = time.time()
        #print("prediction: {}".format(list_pred))

        total_time = end - start
        return {'prediction': list_pred, "prediction_probability": list_pred_prob,
```

Example 23

Project: *transmute-core* Author: *toumorokoshi* File: *example.py* MIT License 5 vc

```
def create_routes_and_handler(transmute_func, context):
    """
    return back a handler that is the api generated
    from the transmute_func, and a list of routes
    it should be mounted to.
    """
    @wraps(transmute_func.raw_func)
    def handler():
        exc, result = None, None
        try:
            args, kwargs = ParamExtractorFlask().extract_params(
                context, transmute_func, request.content_type
            )
            result = transmute_func(*args, **kwargs)
        except Exception as e:
            exc = e
            """
            attaching the traceack is done for you in Python 3, but
            in Python 2 the __traceback__ must be
            attached to the object manually.
            """
            exc.__traceback__ = sys.exc_info()[2]
        """
        transmute_func.process_result handles converting
        the response from the function into the response body,
        the status code that should be returned, and the
        response content-type.
        """
        response = transmute_func.process_result(
            context, result, exc, request.content_type
        )
        return Response(
            response["body"],
            status=response["code"],
            mimetype=response["content-type"],
            headers=response["headers"]
        )
    return (
        _convert_paths_to_flask(transmute_func.paths),
        handler
    )
```

Example 24

Project: *transmute-core* Author: *toumorokoshi* File: *example.py* MIT License 5 vo

```
def add_swagger(app, json_route, html_route, **kwargs):
    """
    add a swagger html page, and a swagger.json generated
    from the routes added to the app.
    """
    spec = getattr(app, SWAGGER_ATTR_NAME)
    if spec:
        spec = spec.swagger_definition(**kwargs)
    else:
        spec = {}
    encoded_spec = json.dumps(spec).encode("UTF-8")

    @app.route(json_route)
    def swagger():
        return Response(
            encoded_spec,
            # we allow CORS, so this can be requested at swagger.io
            headers={"Access-Control-Allow-Origin": "*"},
            content_type="application/json",
        )

    # add the statics
    static_root = get_swagger_static_root()
    swagger_body = generate_swagger_html(
        STATIC_PATH, json_route
    ).encode("utf-8")

    @app.route(html_route)
    def swagger_ui():
        return Response(swagger_body, content_type="text/html")

    # the blueprint work is the easiest way to integrate a static
    # directory into flask.
    blueprint = Blueprint('swagger', __name__, static_url_path=STATIC_PATH,
                          static_folder=static_root)
    app.register_blueprint(blueprint)

# example usage.
```

Example 25

Project: *transmute-core* Author: *toumorokoshi* File: *handler.py* MIT License 5 vo

```
def create_routes_and_handler(transmute_func, context):

    @wraps(transmute_func.raw_func)
    def handler(*args, **kwargs):
        exc, result = None, None
        try:
            args, kwargs = _param_instance.extract_params(
                context, transmute_func, request.content_type,
            )
            result = transmute_func(*args, **kwargs)
        except Exception as e:
            exc = e
            exc.__traceback__ = sys.exc_info()[2]
        response = transmute_func.process_result(
            context, result, exc, request.content_type
        )
        return Response(
            response["body"],
            status=response["code"],
            mimetype=response["content-type"],
            headers=response["headers"],
        )
    return (
        _convert_paths_to_flask(transmute_func.paths),
        handler
    )
```

Example 26

Project: *notify-api* Author: *alphagov* File: *__init__.py* MIT License 5 vo

```
def get_json_from_request(root_element):
    if request.content_type not in [
        'application/json',
        'application/json; charset=UTF-8'
    ]:
        abort(400, "Unexpected Content-Type, expecting 'application/json'")
    data = request.get_json()
    if data is None:
        abort(400, "Invalid JSON; must be a valid JSON object")
    if root_element not in data:
        abort(400, "Invalid JSON; must have {} as root element".format(root_elemer
    return data[root_element]
```

Example 27

Project: *seldon-core* Author: *SeldonIO* File: *flask_utils.py* Apache License 2.0 5 vc

```
def get_request() -> Dict:
    """
    Parse a request to get JSON dict

    Returns
    -------
       JSON Dict

    """

    if (
        request.content_type is not None
        and "multipart/form-data" in request.content_type
    ):
        return get_multi_form_data_request()

    j_str = request.form.get("json")
    if j_str:
        message = json.loads(j_str)
    else:
        j_str = request.args.get("json")
        if j_str:
            message = json.loads(j_str)
        else:
            message = request.get_json()
            if message is None:
                raise SeldonMicroserviceException("Can't find JSON in data")
    if message is None:
        raise SeldonMicroserviceException("Invalid Data Format - empty JSON")
    return message
```

Example 28

Project: *CTFd* Author: *CTFd* File: *__init__.py* Apache License 2.0 5 vc

```
def require_team(f):
    @functools.wraps(f)
    def require_team_wrapper(*args, **kwargs):
        if get_config("user_mode") == TEAMS_MODE:
            team = get_current_team()
            if team is None:
                if request.content_type == "application/json":
                    abort(403)
                else:
                    return redirect(url_for("teams.private", next=request.full_pat
        return f(*args, **kwargs)

    return require_team_wrapper
```

Example 29

Project: *CTFd* Author: *CTFd* File: *visibility.py* Apache License 2.0 5 vc

```
def check_score_visibility(f):
    @functools.wraps(f)
    def _check_score_visibility(*args, **kwargs):
        v = get_config("score_visibility")
        if v == "public":
            return f(*args, **kwargs)
        elif v == "private":
            if authed():
                return f(*args, **kwargs)
            else:
                if request.content_type == "application/json":
                    abort(403)
                else:
                    return redirect(url_for("auth.login", next=request.full_path))

        elif v == "hidden":
            return (
                render_template("errors/403.html", error="Scores are currently hic
                403,
            )

        elif v == "admins":
            if is_admin():
                return f(*args, **kwargs)
            else:
                abort(404)

    return _check_score_visibility
```

Example 30

Project: *store* Author: *Shamilv05* File: *routes.py* MIT License 4 vo

```
def imports():
    if request.content_type != JSON_MIME_TYPE:
        error = json.dumps({'error': 'Invalid Content Type'})
        return json_response(error, 400)

    data = request.json

    try:
        validate(citizens_schema, data)
    except JsonSchemaException as e:
        error = json.dumps({'error': f'{e}'})
        return json_response(error, 400)

    try:
        json_validation(data)
    except ValueError as e:
        error = json.dumps({'error': f'{e}'})
        return json_response(error, 400)
    except KeyError:
        error = json.dumps({'error': 'Relatives array contains nonexistent citizer
        return json_response(error, 400)

    max_import_id_from_table = db.session.query(func.max(Citizen.import_id)).first
    if max_import_id_from_table:
        import_id = max_import_id_from_table + 1
    else:
        import_id = 1

    for item in data['citizens']:
        item.update({"import_id": import_id})

    try:
        db.session.bulk_insert_mappings(Citizen, data['citizens'])
        db.session.commit()
    except exc.SQLAlchemyError:
        db.session.rollback()
        error = json.dumps({'error': 'Cannot insert citizens into db'})
        return json_response(error, 400)

    import_id = {
        "data": {
            "import_id": import_id
        }
    }

    return json_response(json.dumps(import_id))
```

Example 31

Project: *zeus* Author: *getsentry* File: *job_artifacts.py* Apache License 2.0 4 vo

```
def handle_async(self, hook, build_xid, job_xid):
        if request.content_type == "application/json":
            # file must be base64 encoded
            file_data = request.json.get("file")
            if not file_data:
                return self.respond({"file": "Missing file content."}, status=403)

            file = FileStorage(BytesIO(b64decode(file_data)), request.json.get("na
        else:
            try:
                file = request.files["file"]
            except KeyError:
                return self.respond(
                    {"file": "Missing data for required field."}, status=403
                )

        artifact = self.schema_from_request(pending_artifact_schema)
        artifact.external_build_id = build_xid
        artifact.external_job_id = job_xid
        artifact.provider = hook.provider
        artifact.repository_id = hook.repository_id
        artifact.status = Status.queued
        artifact.hook_id = hook.id

        if not artifact.name:
            artifact.name = file.filename

        if not artifact.name:
            return self.respond(
                {"name": "Missing data for required field."}, status=403
            )

        try:
            db.session.add(artifact)
            db.session.flush()
        except IntegrityError:
            db.session.rollback()
            exists = True
        else:
            exists = False

        if exists:
            # XXX(dcramer): this is more of an error but we make an assumption
            # that this happens because it was already sent
            return self.error("An artifact with this name already exists", 204)
        artifact.file.save(
            file,
            "{0}/{1}/{2}_{3}".format(
                job_xid[:4], job_xid[4:], artifact.id.hex, artifact.name
            ),
        )
        db.session.add(artifact)
        db.session.commit()

        return self.respond_with_schema(pending_artifact_schema, artifact, 202)
```

Example 32

Project: *ns4_chatbot* Author: *newsettle* File: *http_server.py* Apache License 2.0 4 vo

```
def _process_web_request(self,flag,func=None):
        data = None
        s_json = None
        if request.content_type is None:#get请求
            data = request.args
        else:#类型为： application/x-www-form-urlencoded
            s_json = request.get_data()
            if s_json is None or s_json == '':
                logger.warn(u"json数据为空")

            else:
                if flag!="coolq_callback":#QQ的回调接口，任何消息都回调，所以不能记录他们
                    logger.debug("接收到来自网络的数据：%s",s_json)

                try:
                    #2018.10.16 bug，文本中有tab\return 旧会导致json解析失败，替换掉
                    s_json = s_json.replace("\t", "").replace("\n", "")
                    # data = json.loads(s_json.decode('ISO-8859-1'))#.decode('ISO-
                    data = json.loads(s_json)

                except ValueError as ve:
                    logger.exception(ve,"无法解析json数据：%s",s_json)
                    return "Error Parse Json: error("+str(ve)+"),json="+s_json,50

        #处理coolq的回调消息，这个是本机上的酷Q-docker接收到消息，回调我们的
        if flag == "coolq_callback":
            rmsg = self.qqbot.on_message(data)
            if rmsg:
                return jsonify({"reply":rmsg})#必须是json格式，文档里说文本即可，不行
            else:
                return "",204#大部分消息不需要处理，直接忽略<https://cqhttp.cc/docs/4.

        bizComponent = self.bizManager.load_biz_comp(flag)

        if bizComponent is None:
            logger.error("无法找到业务组件来为此HTTP请求服务,flag=%s",flag)
            return "系统错误：无法找到内部业务组件给您服务", 500
        else:
            logger.debug("加载业务处理器：%r",bizComponent)

        try:
            result = bizComponent.system2bot(data,func)
        except Exception as ve:
            logger.exception(ve,"业务处理组件发生错误data=[%s],func=[%s]",s_json,func
            return "Error Happen Inside Business Component:"+str(ve),500
        if result=="OK":
            return "OK",200
        else:
            return result, 500
```