Google API Documentation

classroom API Documentation

Version: v1

Manages classes, rosters, and invitations in Google Classroom.

Google API Documentation

Endpoint: registrations.create

HTTP Method: POST

Path: v1/registrations

Description: Creates a `Registration`, causing Classroom to start sending notifications from the provided `feed` to the destination provided in `cloudPubSubTopic`. Returns the created `Registration`. Currently, this will be the same as the argument, but with server-assigned fields such as `expiry_time` and `id` filled in. Note that any value specified for the `expiry_time` or `id` fields will be ignored. While Classroom may validate the `cloudPubSubTopic` and return errors on a best effort basis, it is the caller's responsibility to ensure that it exists and that Classroom has permission to publish to it. This method may return the following error codes: * `PERMISSION_DENIED` if: * the authenticated user does not have permission to receive notifications from the requested field; or * the current user has not granted access to the current Cloud project with the appropriate scope for the requested feed. Note that domain-wide delegation of authority is not currently supported for this purpose. If the request has the appropriate scope, but no grant exists, a Request Errors is returned. * another access error is encountered. * `INVALID_ARGUMENT` if: * no `cloudPubsubTopic` is specified, or the specified `cloudPubsubTopic` is not valid; or * no `feed` is specified, or the specified `feed` is not valid. * `NOT_FOUND` if: * the specified `feed` cannot be located, or the requesting user does not have permission to determine whether or not it exists; or * the specified `cloudPubsubTopic` cannot be located, or Classroom has not been granted permission to publish to it.

AI-Generated Documentation

Technical Description for the POST classroom.registrations.create Endpoint

Common Use Cases:
- Register a student for a classroom.
- Enroll multiple students in a single request.
- Automate the student registration process.

Example Request:
```json
POST /classroom.registrations.create
Content-Type: application/json

{
  "student_id": "12345",
  "classroom_id": "67890",
  "registration_date": "2023-10-01",
  "status": "active"
}
```

Common Parameters:
- **student_id** (string): The unique identifier for the student.
- **classroom_id** (string): The unique identifier for the classroom.
- **registration_date** (string, optional): The date when the registration was made. Format: YYYY-MM-DD.
- **status** (string, optional): The current status of the registration (e.g., "active", "inactive").

Example Code

```python
import requests

def create_classroom_registration(course_id, student_id, access_token):
    """
    Registers a student for a classroom course using the classroom.registrations.create API endpoint.

    Args:
    course_id (str): The ID of the course to register for.
    student_id (str): The ID of the student to register.
    access_token (str): The access token for authentication.

    Returns:
    dict: The response from the API.
    """
    url = 'https://classroom.googleapis.com/v1/courses/{}/students'.format(course_id)
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }
    payload = {
        'studentId': student_id
    }

    try:
        response = requests.post(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an HTTPError for bad responses
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except requests.exceptions.RequestException as err:
        print(f'Error occurred: {err}')
    except Exception as e:
        print(f'An unexpected error occurred: {e}')

# Example usage
course_id = '1234567890'
student_id = '0987654321'
access_token = 'your_access_token_here'

registration_response = create_classroom_registration(course_id, student_id, access_token)
if registration_response:
    print('Registration successful:', registration_response)
```

This code defines a function `create_classroom_registration` that makes a POST request to the `classroom.registrations.create` endpoint to register a student in a classroom course. It includes proper error

```
handling for HTTP errors, request exceptions, and other unexpected errors. The example usage at the bottom shows how to call this function.
```

Google API Documentation

Endpoint: registrations.delete

HTTP Method: DELETE

Path: v1/registrations/{registrationId}

Description: Deletes a `Registration`, causing Classroom to stop sending notifications for that `Registration`.

AI-Generated Documentation

Endpoint: DELETE classroom.registrations.delete

Common Use Cases:
1. Removing a student's registration from a classroom.
2. Deleting a registration entry due to an error or duplicate entry.
3. Cleaning up old or inactive registrations.

Example Request:
```http
DELETE /api/v1/classrooms/{classroomId}/registrations/{registrationId}
```
- Replace `{classroomId}` with the ID of the classroom.
- Replace `{registrationId}` with the ID of the registration to be deleted.

Common Parameters:
1. **classroomId** (Path Parameter): The unique identifier of the classroom from which the registration will be deleted.
2. **registrationId** (Path Parameter): The unique identifier of the registration record that needs to be deleted.

Example:
```http
DELETE /api/v1/classrooms/123/registrations/456
```
- This request deletes the registration with ID `456` from the classroom with ID `123`.

Example Code

```python
import requests

# Define the API endpoint and necessary parameters
url = "https://api.example.com/classroom/registrations/delete"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",  # Replace with your actual access token
    "Content-Type": "application/json"
}
params = {
    "registration_id": "12345",  # Example registration ID
    "student_id": "67890",       # Example student ID
    "course_id": "54321"         # Example course ID
```

```
}

try:
    # Make a DELETE request to the API endpoint
    response = requests.delete(url, headers=headers, json=params)

    # Check if the request was successful
    if response.status_code == 200:
        print("Registration deleted successfully.")
    else:
        print(f"Failed to delete registration. Status code: {response.status_code}")
        print(f"Response: {response.json()}")

except requests.exceptions.RequestException as e:
    # Handle any errors that occur during the request
    print(f"An error occurred: {e}")
```

This Python script demonstrates how to delete a classroom registration using the `classroom.registrations.delete` API endpoint. It includes proper error handling and uses realistic parameter names and values. The script makes a DELETE request to the specified URL with the required headers and parameters. If the request is successful, it prints a success message; otherwise, it prints an error message along with the status code and response. Any exceptions during the request are caught and handled gracefully.

Google API Documentation

Endpoint: invitations.delete

HTTP Method: DELETE

Path: v1/invitations/{id}

Description: Deletes an invitation. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to delete the requested invitation or for access errors. * `NOT_FOUND` if no invitation exists with the requested ID.

AI-Generated Documentation

DELETE classroom.invitations.delete

Common Use Cases:
1. Remove unneeded or expired invitations to a classroom.
2. Clear out invitations for users who have already been added to the classroom.
3. Revoke access to a classroom for users who were mistakenly invited.

Example Request:
```http
DELETE /classroom/invitations/delete HTTP/1.1
Host: api.example.com
Authorization: Bearer your-access-token
Content-Type: application/json

{
  "classroom_id": "12345",
  "invitation_id": "67890"
}
```

Common Parameters:
- **classroom_id**: The unique identifier for the classroom from which the invitation is to be deleted.
- **invitation_id**: The unique identifier for the specific invitation to be deleted.

Example Code

```python
import requests

def delete_invitation(invitation_id, access_token):
    """
    Deletes an invitation using the classroom.invitations.delete API endpoint.

    :param invitation_id: The ID of the invitation to delete.
    :param access_token: The access token for authentication.
    :return: Response from the API.
    """
```

```
    url = "https://classroom.googleapis.com/v1/invitations/{}".format(invitation_id)
    headers = {
        "Authorization": "Bearer {}".format(access_token),
        "Content-Type": "application/json"
    }

    try:
        # Make a DELETE request to the API endpoint
        response = requests.delete(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the response if successful
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
invitation_id = "1234567890abcdef"
access_token = "your_access_token_here"
delete_invitation(invitation_id, access_token)
```

Google API Documentation

Endpoint: invitations.create

HTTP Method: POST

Path: v1/invitations

Description: Creates an invitation. Only one invitation for a user and course may exist at a time. Delete and re-create an invitation to make changes. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to create invitations for this course or for access errors. * `NOT_FOUND` if the course or the user does not exist. * `FAILED_PRECONDITION`: * if the requested user's account is disabled. * if the user already has this role or a role with greater permissions. * for the following request errors: * IneligibleOwner * `ALREADY_EXISTS` if an invitation for the specified user and course already exists.

AI-Generated Documentation

API Endpoint: POST classroom.invitations.create

Common Use Cases:
- Inviting a new user to join a classroom.
- Sending multiple invitations to a group of users.
- Re-issuing a classroom invitation if the original invitation was lost or expired.

Example Request:
```http
POST https://api.example.com/classroom.invitations.create
Content-Type: application/json

{
  "classroom_id": "12345",
  "invitee_email": "invitee@example.com",
  "message": "You are invited to join our classroom!"
}
```

Common Parameters:
- **classroom_id** (required): The unique identifier for the classroom where the invitation is being sent.
- **invitee_email** (required): The email address of the user being invited.
- **message** (optional): A custom message to include with the invitation.
- **expires_at** (optional): The timestamp indicating when the invitation will expire.

Example Code

```python
import requests

def create_classroom_invitation(token, email, class_id, role):
    """
    Creates a classroom invitation using the classroom.invitations.create API endpoint.
```

```
    :param token: The authentication token for the API.
    :param email: The email address of the user to invite.
    :param class_id: The ID of the class to invite the user to.
    :param role: The role to assign to the invited user (e.g., 'STUDENT' or 'TEACHER').
    :return: The response from the API, or None in case of an error.
    """
    url = "https://classroom.googleapis.com/v1/invitations"
    headers = {
        "Authorization": f"Bearer {token}",
        "Content-Type": "application/json"
    }
    data = {
        "email": email,
        "classId": class_id,
        "role": role
    }

    try:
        response = requests.post(url, headers=headers, json=data)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
    except Exception as e:
        print(f"An error occurred: {e}")
    return None

# Example usage
token = "your_auth_token_here"
email = "invitee@example.com"
class_id = "1234567890"
role = "STUDENT"

invitation = create_classroom_invitation(token, email, class_id, role)
if invitation:
    print("Invitation created successfully:", invitation)
else:
    print("Failed to create invitation.")
```

Google API Documentation

Endpoint: invitations.get

HTTP Method: GET

Path: v1/invitations/{id}

Description: Returns an invitation. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to view the requested invitation or for access errors. * `NOT_FOUND` if no invitation exists with the requested ID.

AI-Generated Documentation

Common Use Cases:
- Retrieving a list of invitations for a classroom.
- Checking the status of outstanding invitations.
- Verifying pending invitations for user acceptance.

Example Request:
```
GET /api/v1/classroom/invitations/get
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **classroom_id** (string, required): The unique identifier for the classroom.
- **status** (string, optional): Filter invitations by status (e.g., pending, accepted, rejected).
- **user_id** (string, optional): Filter invitations by a specific user ID.
- **limit** (integer, optional): The maximum number of invitations to return (default: 10).
- **offset** (integer, optional): The number of invitations to skip (default: 0).

Example Code

```python
import requests

def get_classroom_invitations(course_id, user_id, token):
    """
    Fetches invitations for a classroom using the classroom.invitations.get endpoint.

    Parameters:
    course_id (str): The ID of the course.
    user_id (str): The ID of the user.
    token (str): The authentication token.

    Returns:
    dict: The JSON response from the API.
    """
```

```
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/invitations"

    headers = {
        "Authorization": f"Bearer {token}"
    }

    params = {
        "userId": user_id
    }

    try:
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")  # Log the error to help with debugging
    except Exception as err:
        print(f"An error occurred: {err}")  # Log any other errors
    return None

# Example usage:
course_id = "1234567890"
user_id = "user123"
token = "ya29.AHES6ZS... restofthetoken"

invitations = get_classroom_invitations(course_id, user_id, token)
if invitations:
    print("Invitations:", invitations)
else:
    print("Failed to retrieve invitations.")
```

Google API Documentation

Endpoint: invitations.list

HTTP Method: GET

Path: v1/invitations

Description: Returns a list of invitations that the requesting user is permitted to view, restricted to those that match the list request. *Note:* At least one of `user_id` or `course_id` must be supplied. Both fields can be supplied. This method returns the following error codes: * `PERMISSION_DENIED` for access errors.

AI-Generated Documentation

API Endpoint: GET classroom.invitations.list

Common Use Cases:
1. **List Pending Invitations**: Administrators can use this endpoint to list all pending invitations to classrooms.
2. **Monitor Classroom Activity**: Educators can monitor who has been invited to join their classrooms.
3. **Invitation Management**: Users can check the status of their own invitations to classrooms.

Example Request:
```
GET https://api.example.com/classroom.invitations.list?userId=12345&status=pending
```

Common Parameters:
- `userId` (optional): The ID of the user whose invitations are to be listed.
- `status` (optional): The status of the invitations to filter (e.g., "pending", "accepted", "declined").
- `classroomId` (optional): The ID of the classroom whose invitations are to be listed.
- `limit` (optional): The maximum number of invitations to return.
- `offset` (optional): The number of invitations to skip before starting to collect the result set.

Example Code

```python
import requests

def list_classroom_invitations():
    # API endpoint URL
    url = "https://classroom.googleapis.com/v1/courses/{course_id}/invitations"

    # Realistic parameter values
    course_id = "123456789"  # Replace with a real course ID
    params = {
        'pageSize': 10,  # Number of invitations to return per page
        'pageToken': 'abc123'  # Token for pagination (if any)
    }

    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN'  # Replace with a valid access token
```

```
    }

    try:
        # Make the API request
        response = requests.get(url, headers=headers, params=params)

        # Check for HTTP errors
        response.raise_for_status()

        # Parse the JSON response
        invitations = response.json()

        # Print the invitations (or process as needed)
        for invitation in invitations.get('invitations', []):
            print(invitation)

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Error occurred: {req_err}")
    except ValueError as json_err:
        print(f"JSON decode error: {json_err}")

# Call the function to list classroom invitations
list_classroom_invitations()
```

Google API Documentation

Endpoint: invitations.accept

HTTP Method: POST

Path: v1/invitations/{id}:accept

Description: Accepts an invitation, removing it and adding the invited user to the teachers or students (as appropriate) of the specified course. Only the invited user may accept an invitation. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to accept the requested invitation or for access errors. * `FAILED_PRECONDITION` for the following request errors: * CourseMemberLimitReached * CourseNotModifiable * CourseTeacherLimitReached * UserGroupsMembershipLimitReached * `NOT_FOUND` if no invitation exists with the requested ID.

AI-Generated Documentation

Technical Description for POST classroom.invitations.accept

Common Use Cases:
- Accepting a pending invitation to join a classroom.
- Automating user enrollment in a classroom via an application.

Example Request:
```http
POST /classroom/invitations/accept
Content-Type: application/json

{
  "invitationId": "12345",
  "userId": "user67890"
}
```

Common Parameters:
- `invitationId` (string): The unique identifier for the invitation to be accepted.
- `userId` (string): The unique identifier for the user accepting the invitation.

Example Code

```python
import requests

def accept_classroom_invitation(invitation_id, user_id, access_token):
    """
    Accepts a classroom invitation using the provided invitation ID, user ID, and access token.

    :param invitation_id: The ID of the invitation to accept.
    :param user_id: The ID of the user accepting the invitation.
    :param access_token: The OAuth 2.0 access token.
    :return: Response from the API.
```

```
    """

    url = "https://classroom.googleapis.com/v1/invitations/{}/accept".format(invitation_id)
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }

    try:
        response = requests.post(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)

        # Check if the response contains an error
        if 'error' in response.json():
            raise ValueError(response.json()['error']['message'])

        return response.json()  # Return the JSON response from the API

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")  # HTTP error
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")  # Connection error
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")  # Timeout error
    except requests.exceptions.RequestException as req_err:
        print(f"An error occurred: {req_err}")  # Generic error
    except ValueError as val_err:
        print(f"Value error occurred: {val_err}")  # Value error for JSON response

# Example usage
invitation_id = "1234567890"
user_id = "user123"
access_token = "your_oauth2_access_token"

response = accept_classroom_invitation(invitation_id, user_id, access_token)
if response:
    print("Invitation accepted successfully:", response)
```

This code defines a function `accept_classroom_invitation` that accepts a classroom invitation by sending a POST request to the specified endpoint. It includes error handling for different types of exceptions that might occur during the HTTP request and response processing. The example usage shows how to call this function with realistic parameter values.

Google API Documentation

Endpoint: courses.update

HTTP Method: PUT

Path: v1/courses/{id}

Description: Updates a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to modify the requested course or for access errors. * `NOT_FOUND` if no course exists with the requested ID. * `FAILED_PRECONDITION` for the following request errors: * CourseNotModifiable

AI-Generated Documentation

Technical Description for PUT classroom.courses.update

Common Use Cases
- Update the course details such as title, description, or schedule.
- Modify the instructor or teacher associated with a course.
- Adjust the enrollment status or prerequisites for a course.

Example Request

```http
PUT /api/classroom/courses/12345
Content-Type: application/json

{
  "title": "Advanced Mathematics",
  "description": "An in-depth study of advanced mathematical concepts.",
  "instructor": "Dr. John Doe",
  "schedule": "MWF 10-11 AM",
  "prerequisites": ["Basic Mathematics"]
}
```

Common Parameters

- **course_id** (string): The unique identifier for the course to be updated.
- **title** (string): The new title for the course.
- **description** (string): A detailed description of the course content.
- **instructor** (string): The name or ID of the instructor for the course.
- **schedule** (string): The updated schedule for the course.
- **prerequisites** (array of strings): A list of prerequisites for the course.

Example Code

```python
import requests

def update_course(course_id, name, section, description, teacher_email):
```

```
    url = "https://classroom.googleapis.com/v1/courses/" + course_id
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }
    payload = {
        "name": name,
        "section": section,
        "description": description,
        "ownerId": teacher_email
    }

    try:
        response = requests.put(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
    except requests.exceptions.JSONDecodeError:
        print("Error decoding JSON response.")

# Example usage:
course_id = "course123"
name = "Advanced Mathematics"
section = "101"
description = "This course covers advanced topics in mathematics."
teacher_email = "teacher@example.com"

updated_course = update_course(course_id, name, section, description, teacher_email)
print(updated_course)
```

Google API Documentation

Endpoint: courses.get

HTTP Method: GET

Path: v1/courses/{id}

Description: Returns a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or for access errors. * `NOT_FOUND` if no course exists with the requested ID.

AI-Generated Documentation

Common Use Cases:
- Retrieving a list of courses associated with a classroom.
- Fetching course details for display in a user interface.
- Populating a course selection dropdown in an application.

Example Request:
```http
GET https://api.example.com/classroom/courses/get?classroomId=12345
```

Common Parameters:
- **classroomId**: (Required) The unique identifier for the classroom to retrieve courses from.
- **page**: (Optional) The page number for pagination. Default is 1.
- **limit**: (Optional) The number of courses to return per page. Default is 10.
- **sortBy**: (Optional) The field to sort the courses by. Common values include 'name' and 'date'.
- **order**: (Optional) The sort order. Values are 'asc' for ascending and 'desc' for descending. Default is 'asc'.

Example Code

```python
import requests

def get_course_details(course_id, access_token):
    """
    Fetches course details from the classroom.courses.get API.

    :param course_id: The ID of the course to fetch.
    :param access_token: The OAuth 2.0 access token.
    :return: Course details if successful, otherwise an error message.
    """
    # Define the API endpoint and headers
    url = "https://classroom.googleapis.com/v1/courses"
    headers = {
        "Authorization": f"Bearer {access_token}"
    }
    # Define the parameters for the request
    params = {
```

```
        "id": course_id
    }

    try:
        # Make the API request
        response = requests.get(f"{url}/{course_id}", headers=headers, params=params)

        # Check if the request was successful
        if response.status_code == 200:
            # Parse the JSON response
            course_details = response.json()
            return course_details
        else:
            # Return an error message if the request failed
            return f"Error: {response.status_code} - {response.text}"
    except requests.exceptions.RequestException as e:
        # Handle any request exceptions
        return f"Request failed: {e}"

# Example usage
course_id = "123456789"  # Replace with a real course ID
access_token = "your_access_token_here"  # Replace with a real access token

course_details = get_course_details(course_id, access_token)
print(course_details)
```

Google API Documentation

Endpoint: courses.create

HTTP Method: POST

Path: v1/courses

Description: Creates a course. The user specified in `ownerId` is the owner of the created course and added as a teacher. A non-admin requesting user can only create a course with themselves as the owner. Domain admins can create courses owned by any user within their domain. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to create courses or for access errors. * `NOT_FOUND` if the primary teacher is not a valid user. * `FAILED_PRECONDITION` if the course owner's account is disabled or for the following request errors: * UserCannotOwnCourse * UserGroupsMembershipLimitReached * `ALREADY_EXISTS` if an alias was specified in the `id` and already exists.

AI-Generated Documentation

Endpoint: POST /classroom/courses/create

Common Use Cases:
- Creating a new course in an educational platform.
- Registering a course for a specific academic term.
- Enabling administrators to set up new courses with specific details.

Example Request:
```json
{
  "name": "Introduction to Computer Science",
  "description": "A foundational course in computer science principles",
  "instructor_id": "12345",
  "term": "Fall 2023",
  "start_date": "2023-09-01",
  "end_date": "2023-12-15"
}
```

Common Parameters:
- **name**: (String) The name of the course.
- **description**: (String) A brief description of the course content.
- **instructor_id**: (String) The unique identifier for the instructor.
- **term**: (String) The academic term for which the course is offered (e.g., "Fall 2023").
- **start_date**: (Date) The start date of the course.
- **end_date**: (Date) The end date of the course.

Example Code

```python
import requests

# Define the API endpoint and headers
```

```
url = "https://classroom.googleapis.com/v1/courses"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the payload with realistic parameter names and values
payload = {
    "name": "Introduction to Python Programming",
    "section": "CS101",
    "descriptionHeading": "Learn Python from scratch",
    "description": "This course covers the basics of Python programming.",
    "room": "Room 101",
    "ownerId": "user12345",
    "courseState": "PROVISIONED",
    "enrollmentCode": "ABC123",
    "creationTime": "2023-10-01T12:00:00.000Z"
}

# Make the POST request to create a course
response = requests.post(url, headers=headers, json=payload)

# Check for HTTP errors
if response.status_code == 200:
    # Successfully created the course
    course_data = response.json()
    print("Course created successfully:", course_data)
else:
    # Handle errors
    print("Failed to create course. Status code:", response.status_code)
    error_data = response.json()
    print("Error details:", error_data)
```

Google API Documentation

Endpoint: courses.list

HTTP Method: GET

Path: v1/courses

Description: Returns a list of courses that the requesting user is permitted to view, restricted to those that match the request. Returned courses are ordered by creation time, with the most recently created coming first. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the query argument is malformed. * `NOT_FOUND` if any users specified in the query arguments do not exist.

AI-Generated Documentation

Endpoint: GET classroom.courses.list

Common Use Cases:
- Retrieving a list of all courses available in a classroom.
- Fetching course details for administrative purposes.
- Assisting students in browsing available courses.

Example Request:
```
GET https://api.example.com/classroom/courses/list?classroomId=12345&page=1&limit=10
```

Common Parameters:
- **classroomId** (string, required): Identifier for the specific classroom.
- **page** (integer, optional): The page number to retrieve, defaults to 1.
- **limit** (integer, optional): The number of courses to return per page, defaults to 10.
- **sort** (string, optional): The field to sort the courses by (e.g., "name", "startDate").
- **filter** (string, optional): A query string to filter courses (e.g., "status=active").

Example Code

```python
import requests

def list_courses(api_key, course_state='ACTIVE', page_token=None):
    # API endpoint URL
    url = 'https://classroom.googleapis.com/v1/courses'

    # Parameters for the request
    params = {
        'key': api_key,  # Your API key
        'state': course_state,  # State of the course (e.g., ACTIVE, ARCHIVED)
        'pageToken': page_token  # Token for pagination
    }

    try:
```

```
        # Make the GET request to the API
        response = requests.get(url, params=params)
        response.raise_for_status()  # Raise an exception for HTTP errors

        # Parse the JSON response
        courses = response.json()

        # Return the list of courses
        return courses

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # Print HTTP error
    except Exception as err:
        print(f'Other error occurred: {err}')  # Print any other error

    # Return None if an error occurs
    return None

# Example usage
api_key = 'YOUR_API_KEY_HERE'
courses = list_courses(api_key)
print(courses)
```

Google API Documentation

Endpoint: courses.patch

HTTP Method: PATCH

Path: v1/courses/{id}

Description: Updates one or more fields in a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to modify the requested course or for access errors. * `NOT_FOUND` if no course exists with the requested ID. * `INVALID_ARGUMENT` if invalid fields are specified in the update mask or if no update mask is supplied. * `FAILED_PRECONDITION` for the following request errors: * CourseNotModifiable * InactiveCourseOwner * IneligibleOwner

AI-Generated Documentation

Technical Description

Endpoint: `PATCH classroom.courses.patch`

Common Use Cases:
- Update specific fields of an existing course.
- Modify course details such as title, description, or start/end dates.
- Adjust course settings or configurations.

Example Request:
```http
PATCH /classroom/courses/{course_id}
Content-Type: application/json

{
  "title": "New Course Title",
  "description": "Updated course description.",
  "start_date": "2023-12-01",
  "end_date": "2024-06-30"
}
```

Common Parameters:
- `course_id` (Path): The unique identifier of the course to be updated.
- `title` (Body): The updated title of the course.
- `description` (Body): The updated description of the course.
- `start_date` (Body): The new start date of the course in `YYYY-MM-DD` format.
- `end_date` (Body): The new end date of the course in `YYYY-MM-DD` format.
- `settings` (Body): Additional settings or configurations for the course, usually in JSON format.

Example Code

```python
import requests
```

```
# Define the API endpoint and headers
url = "https://classroom.googleapis.com/v1/courses"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the course ID and the data to be updated
course_id = "course123456"
update_data = {
    "id": course_id,
    "name": "Updated Course Name",
    "section": "Updated Section",
    "descriptionHeading": "Updated Description Heading",
    "description": "Updated Course Description"
}

# Define the request URL with the course ID
request_url = f"{url}/{course_id}"

# Make the PATCH request to update the course
response = requests.patch(request_url, headers=headers, json=update_data)

# Check for HTTP errors
if response.status_code == 200:
    print("Course updated successfully!")
    print("Response:", response.json())
elif response.status_code == 400:
    print("Bad request: Check the request parameters.")
    print("Response:", response.json())
elif response.status_code == 401:
    print("Unauthorized: Check your access token.")
    print("Response:", response.json())
elif response.status_code == 403:
    print("Forbidden: You may not have the necessary permissions.")
    print("Response:", response.json())
elif response.status_code == 404:
    print("Not found: The course ID may be incorrect.")
    print("Response:", response.json())
else:
    print(f"An error occurred: {response.status_code}")
    print("Response:", response.json())
```

Google API Documentation

Endpoint: courses.delete

HTTP Method: DELETE

Path: v1/courses/{id}

Description: Deletes a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to delete the requested course or for access errors. * `NOT_FOUND` if no course exists with the requested ID.

AI-Generated Documentation

Technical Description for DELETE classroom.courses.delete

Common Use Cases:
- Removing a course from the classroom schedule.
- Deleting a course that is no longer relevant or needed.
- Freeing up resources associated with a course.

Example Request:
```http
DELETE /classroom/courses/delete HTTP/1.1
Host: api.example.com
Content-Type: application/json
Authorization: Bearer YOUR_ACCESS_TOKEN

{
  "course_id": "12345",
  "user_id": "67890"
}
```

Common Parameters:
- **course_id** (string, required): The unique identifier for the course to be deleted.
- **user_id** (string, required): The unique identifier for the user initiating the deletion.
- **Authorization** (string, required): The access token for authentication.

Google API Documentation

Endpoint: courses.aliases.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/aliases/{alias}

Description: Deletes an alias of a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to remove the alias or for access errors. * `NOT_FOUND` if the alias does not exist. * `FAILED_PRECONDITION` if the alias requested does not make sense for the requesting user or course (for example, if a user not in a domain attempts to delete a domain-scoped alias).

AI-Generated Documentation

Technical Description for DELETE classroom.courses.aliases.delete

Common Use Cases:
- Remove an alias from a course.
- Cleanup unused or outdated aliases.
- Ensure data consistency by deleting duplicates.

Example Request:
```http
DELETE /classroom/courses/{course_id}/aliases/{alias_id}
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```
- Replace `{course_id}` with the ID of the course.
- Replace `{alias_id}` with the ID of the alias to be deleted.

Common Parameters:
- **course_id** (Path Parameter, Required): The unique identifier of the course from which the alias will be deleted.
- **alias_id** (Path Parameter, Required): The unique identifier of the alias to be deleted.
- **Authorization** (Header, Required): The Bearer token for authentication.

Example Code

```python
import requests

def delete_course_alias(alias_id, course_id, access_token):
    """
    Deletes a course alias using the classroom.courses.aliases.delete API endpoint.

    Parameters:
    alias_id (str): The ID of the alias to delete.
    course_id (str): The ID of the course.
    access_token (str): The access token for authentication.

    Returns:
```

```
    dict: The response from the API.
    """

    # Define the API endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/aliases/{alias_id}"

    # Define the headers with the access token
    headers = {
        'Authorization': f'Bearer {access_token}'
    }

    try:
        # Send the DELETE request
        response = requests.delete(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the response JSON
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # Print HTTP errors
    except Exception as err:
        print(f'Other error occurred: {err}')  # Print other errors

    return None  # Return None if an error occurs

# Example usage
alias_id = "example_alias_id"
course_id = "example_course_id"
access_token = "your_access_token_here"

response = delete_course_alias(alias_id, course_id, access_token)
if response:
    print("Alias deleted successfully:", response)
else:
    print("Failed to delete alias.")
```

Google API Documentation

Endpoint: courses.aliases.list

HTTP Method: GET

Path: v1/courses/{courseId}/aliases

Description: Returns a list of aliases for a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the course or for access errors. * `NOT_FOUND` if the course does not exist.

AI-Generated Documentation

Common Use Cases
- Retrieve all alias names for a specific course.
- Validate if a course has any aliases.
- Retrieve alias details for integration and reporting purposes.

Example Request
```http
GET /classroom.courses.aliases.list?course_id=12345&limit=10&offset=0
```

Common Parameters
- **course_id**: (Required) The unique identifier for the course.
- **limit**: (Optional) The maximum number of aliases to return.
- **offset**: (Optional) The number of aliases to skip before starting to return the results.

Example Code

```python
import requests

def list_course_aliases(course_id, access_token):
    """
    Fetch the list of course aliases for a given course ID.

    Parameters:
    - course_id (str): The ID of the course.
    - access_token (str): The access token for authentication.

    Returns:
    - list: A list of course aliases, or None if the request fails.
    """
    url = "https://classroom.googleapis.com/v1/courses/{}/aliases".format(course_id)
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }
```

```
    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        aliases = response.json().get('aliases', [])
        return aliases
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")
    return None

# Example usage:
course_id = "course_12345"  # Replace with an actual course ID
access_token = "your_access_token_here"  # Replace with an actual access token

aliases = list_course_aliases(course_id, access_token)
if aliases is not None:
    print("Course Aliases:", aliases)
else:
    print("Failed to retrieve course aliases.")
```

Google API Documentation

Endpoint: courses.aliases.create

HTTP Method: POST

Path: v1/courses/{courseId}/aliases

Description: Creates an alias for a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to create the alias or for access errors. * `NOT_FOUND` if the course does not exist. * `ALREADY_EXISTS` if the alias already exists. * `FAILED_PRECONDITION` if the alias requested does not make sense for the requesting user or course (for example, if a user not in a domain attempts to access a domain-scoped alias).

AI-Generated Documentation

Endpoint: POST classroom.courses.aliases.create

Common Use Cases
- Creating an alias for a course to facilitate easier access or identification.
- Mitigating issues with long or complex course names by providing a more user-friendly alias.
- Enhancing the organization and navigation within a course management system.

Example Request
```json
{
  "course_id": "12345",
  "alias": "Math101_2023"
}
```

Common Parameters
- **course_id**: (String) Unique identifier for the course for which the alias is being created.
- **alias**: (String) The alias name to be assigned to the course.

Google API Documentation

Endpoint: courses.posts.getAddOnContext

HTTP Method: GET

Path: v1/courses/{courseId}/posts/{postId}/addOnContext

Description: Gets metadata for Classroom add-ons in the context of a specific post. To maintain the integrity of its own data and permissions model, an add-on should call this to validate query parameters and the requesting user's role whenever the add-on is opened in an [iframe](https://developers.google.com/classroom/add-ons/get-started/iframes/iframes-overview). This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.posts.getAddOnContext

Common Use Cases:
- Retrieve additional context or metadata for a specific post within a course.
- Enrich course posts with supplementary information for better user experience.
- Support integrations with third-party applications that require additional context about course posts.

Example Request:
```
GET /classroom/courses/123/posts/456/getAddOnContext
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId** (required): The unique identifier for the course.
- **postId** (required): The unique identifier for the post within the course.
- **accessToken** (required, header): The authorization token to validate the request.

Example Code

```python
import requests

def get_add_on_context(course_id, post_id, access_token):
    """
    Function to get the add-on context for a specific post in a course.

    :param course_id: ID of the course
    :param post_id: ID of the post
    :param access_token: Access token for authentication
    :return: JSON response from the API
    """
```

```
    # Define the URL for the API endpoint
    url = "https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/getAddOnContext".format(
        course_id=course_id,
        post_id=post_id
    )

    # Define the headers including the authorization token
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }

    try:
        # Make the GET request to the API endpoint
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        response.raise_for_status()

        # Return the JSON response
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Error occurred: {req_err}")
    except Exception as err:
        print(f"An unexpected error occurred: {err}")

    # Return None if an error occurred
    return None

# Example usage
course_id = "1234567890"
post_id = "0987654321"
access_token = "your_access_token_here"

context = get_add_on_context(course_id, post_id, access_token)
if context:
    print("Add-on context:", context)
else:
    print("Failed to retrieve add-on context.")
```

This code defines a function `get_add_on_context` that makes a GET request to the

Google API Documentation

`classroom.courses.posts.getAddOnContext` API endpoint. It includes proper error handling for various types of request exceptions and returns the JSON response if successful.

Google API Documentation

Endpoint: courses.posts.addOnAttachments.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments/{attachmentId}

Description: Updates an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Endpoint: PATCH classroom.courses.posts.addOnAttachments.patch

Common Use Cases:
- Update the attachments for a specific post within a course.
- Modify metadata or properties of an existing attachment.
- Add new attachments to an existing post.

Example Request:

```http
PATCH /classroom/courses/123/posts/456/addOnAttachments HTTP/1.1
Host: api.example.com
Content-Type: application/json
Authorization: Bearer YOUR_ACCESS_TOKEN

{
  "attachments": [
    {
      "id": "attachment1",
      "url": "https://example.com/files/attachment1.png",
      "name": "UpdatedAttachment1",
      "description": "Updated description for attachment 1"
    },
    {
      "id": "attachment2",
      "url": "https://example.com/files/attachment2.pdf",
      "name": "Attachment 2",
      "description": "Description for attachment 2"
    }
  ]
}
```

Common Parameters:

- `attachments` (Array of Objects): An array of attachment objects to be updated or added.
  - `id` (String): The unique identifier of the attachment.

- `url` (String): The URL of the attachment file.
- `name` (String, Optional): The name of the attachment.
- `description` (String, Optional): A description of the attachment.

Example Code

```python
import requests

# Define the API endpoint and the necessary parameters
url = "https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments/{attachment_id}"
course_id = "1234567890"
post_id = "1234567890"
attachment_id = "1234567890"
token = "YOUR_ACCESS_TOKEN"  # Replace with your access token

# Define the headers including the authorization token
headers = {
    "Authorization": f"Bearer {token}",
    "Content-Type": "application/json"
}

# Define the data payload for the PATCH request
data = {
    "fileId": "new_file_id",  # The new file ID to attach
    "title": "Updated Attachment Title",  # Title of the attachment
    "description": "Updated description for the attachment"  # Description of the attachment
}

# Make the PATCH request to update the attachment
try:
    response = requests.patch(url, headers=headers, json=data)
    response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
    print("Attachment updated successfully:", response.json())
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")  # Handle HTTP errors
except requests.exceptions.RequestException as req_err:
    print(f"An error occurred: {req_err}")  # Handle other request exceptions
```

This code demonstrates how to make a PATCH request to the `classroom.courses.posts.addOnAttachments.patch` endpoint with proper error handling and realistic parameter names and values. Adjust the `url`, `course_id`, `post_id`, `attachment_id`, and `token` to match your specific use case.

Google API Documentation

Endpoint: courses.posts.addOnAttachments.create

HTTP Method: POST

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments

Description: Creates an add-on attachment under a post. Requires the add-on to have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for POST classroom.courses.posts.addOnAttachments.create

Common Use Cases
- Adding attachments to a post within a course.
- Uploading supplementary materials such as PDFs, images, or documents to enhance course content.
- Enabling instructors to provide additional resources for students.

Example Request
```http
POST /classroom/courses/123/posts/456/addOnAttachments/create
Content-Type: multipart/form-data

--boundary
Content-Disposition: form-data; name="file"; filename="example.pdf"
Content-Type: application/pdf

--boundary--
```

Common Parameters
- **courseId** (integer, required): The unique identifier for the course to which the post belongs.
- **postId** (integer, required): The unique identifier for the post to which the attachment will be added.
- **file** (file, required): The file to be uploaded as an attachment. Supported formats include PDF, JPEG, PNG, and DOCX.

These parameters are typically included in the form-data of the request.

Example Code

```python
import requests

# Define the API endpoint and the required parameters
api_url = "https://classroom.googleapis.com/v1/courses/{courseId}/posts/{postId}/attachments"

# Replace with realistic values
course_id = "1234567890"
```

```
post_id = "9876543210"
file_id = "file_id_example"
title = "Sample Attachment"

# Set up the headers for the request (e.g., authorization token)
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"
}

# Prepare the request payload
payload = {
    "fileId": file_id,
    "title": title
}

try:
    # Send the POST request to add an attachment to a post
    response = requests.post(f"{api_url}?courseId={course_id}&postId={post_id}", headers=headers, json=payload)

    # Check the response status code
    if response.status_code == 200:
        print("Attachment added successfully.")
        print("Response:", response.json())
    else:
        print(f"Failed to add attachment. Status code: {response.status_code}")
        print("Response:", response.json())
except requests.exceptions.RequestException as e:
    # Handle any requests exceptions
    print(f"An error occurred: {e}")
```

Google API Documentation

Endpoint: courses.posts.addOnAttachments.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments/{attachmentId}

Description: Deletes an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint: DELETE classroom.courses.posts.addOnAttachments.delete

Common Use Cases:
1. **Removing Attachments:** Delete specific attachments added to a course post.
2. **Cleanup:** Ensure that outdated or irrelevant attachments are removed from the system.

Example Request:
```
DELETE /classroom/courses/:courseId/posts/:postId/addOnAttachments/:attachmentId
Authorization: Bearer <token>
```

Common Parameters:
- **courseId:** The unique identifier of the course.
- **postId:** The unique identifier of the post within the course.
- **attachmentId:** The unique identifier of the attachment to be deleted.

Example Code

```python
import requests

def delete_course_post_attachment(course_id, post_id, attachment_id, access_token):
    """
    Deletes an attachment from a specific post in a course.

    :param course_id: ID of the course.
    :param post_id: ID of the post.
    :param attachment_id: ID of the attachment to delete.
    :param access_token: Access token for authentication.
    :return: Response from the API.
    """
                                                                                url                     =
f"https://api.classroom.example.com/v1/courses/{course_id}/posts/{post_id}/attachments/{attachment_id}"

    headers = {
        'Authorization': f'Bearer {access_token}',
```

```
        'Content-Type': 'application/json'
    }

    try:
        response = requests.delete(url, headers=headers)
        response.raise_for_status()  # Raise HTTPError for bad responses (4xx or 5xx)
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

# Example usage:
course_id = 12345
post_id = 67890
attachment_id = 111213
access_token = "your_access_token_here"

delete_course_post_attachment(course_id, post_id, attachment_id, access_token)
```

Google API Documentation

Endpoint: courses.posts.addOnAttachments.get

HTTP Method: GET

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments/{attachmentId}

Description: Returns an add-on attachment. Requires the add-on requesting the attachment to be the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Endpoint: GET classroom.courses.posts.addOnAttachments.get

Common Use Cases:
- Fetching attachments for a specific course post.
- Verifying the availability and details of attachments for a course post.
- Retrieving metadata about attachments to be displayed in the user interface.

Example Request:

```http
GET /classroom/courses/{courseId}/posts/{postId}/addOnAttachments?contentType=pdf&limit=10
```

Common Parameters:
- `courseId` (string, required): The unique identifier for the course.
- `postId` (string, required): The unique identifier for the post within the course.
- `contentType` (string, optional): Filters the attachments by content type (e.g., pdf, doc, image).
- `limit` (integer, optional): Limits the number of attachments returned in the response.
- `offset` (integer, optional): Specifies the starting point for the attachments to return.

Example Code

```python
import requests

# Define the API endpoint and parameters
url = "https://classroom.googleapis.com/v1/courses/{courseId}/posts/{postId}/addOnAttachments"
params = {
    "courseId": "1234567890",
    "postId": "8901234567",
    "attachments": [
        {
            "title": "Homework Assignment",
            "fileUrl": "https://example.com/homework.pdf"
        }
    ]
}
```

```
# Set up the headers for the request
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

try:
    # Make the POST request to add attachments
    response = requests.post(url, headers=headers, json=params)

    # Check if the request was successful
    response.raise_for_status()

    # Print the response from the API
    print("Attachments added successfully:", response.json())

except requests.exceptions.HTTPError as http_err:
    # Handle HTTP errors
    print(f"HTTP error occurred: {http_err}")

except requests.exceptions.RequestException as err:
    # Handle other errors
    print(f"An error occurred: {err}")

except Exception as e:
    # Handle any other exceptions
    print(f"An unexpected error occurred: {e}")
```

Google API Documentation

Endpoint: courses.posts.addOnAttachments.list

HTTP Method: GET

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments

Description: Returns all attachments created by an add-on under the post. Requires the add-on to have active attachments on the post or have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Endpoint: GET classroom.courses.posts.addOnAttachments.list

Common Use Cases
- Retrieve a list of attachments added to a specific post within a course.
- Verify the status or details of attachments for a particular course post.
- Integrate with educational platforms to manage and display course materials dynamically.

Example Request
```
GET /classroom.courses.posts.addOnAttachments.list?courseId=12345&postId=67890
```

Common Parameters
- **courseId** (required): The unique identifier for the course.
- **postId** (required): The unique identifier for the post within the course.
- **pageToken** (optional): A token to retrieve the next set of results in paginated responses.
- **maxResults** (optional): The maximum number of results to return in a single response (default: 10).

Example Code

```python
import requests

# Define the API endpoint URL
url = "https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments"

# Define the parameters for the request
params = {
    'fields': 'attachments(id,title,url)',
    'pageSize': 10
}

# Define the headers for the request, including the Authorization token
headers = {
    'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
    'Content-Type': 'application/json'
```

```
}

# Function to get attachments for a post in a Google Classroom course
def get_post_attachments(course_id, post_id):
    try:
        # Make the API request
                response  =  requests.get(url.format(course_id=course_id,  post_id=post_id),  headers=headers,
params=params)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Return the list of attachments
        return data.get('attachments', [])

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except requests.exceptions.RequestException as req_err:
        print(f'Error occurred: {req_err}')
    except Exception as err:
        print(f'An error occurred: {err}')

# Example usage
course_id = '1234567890'
post_id = '0987654321'
attachments = get_post_attachments(course_id, post_id)
print(attachments)
```

This code defines a function `get_post_attachments` that makes a GET request to the Google Classroom API to retrieve attachments for a specific post in a course. It includes proper error handling for HTTP errors, request exceptions, and general exceptions. The function returns a list of attachments if successful or prints an error message if an exception occurs.

Google API Documentation

Endpoint: courses.posts.addOnAttachments.studentSubmissions.get

HTTP Method: GET

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments/{attachmentId}/studentSubmissions/{submissionId}

Description: Returns a student submission for an add-on attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.posts.addOnAttachments.studentSubmissions.get

Common Use Cases:
- Retrieve a list of student submissions for a specific assignment post with attached files.
- Monitor and verify student progress by checking their submission status and attachments.
- Automate grading processes by accessing submission files programmatically.

Example Request:
```http
GET https://api.example.com/v1/classroom/courses/{courseId}/posts/{postId}/addOnAttachments/studentSubmissions
```

Common Parameters:
- `courseId` (path, required): The unique identifier for the course.
- `postId` (path, required): The unique identifier for the assignment post.
- `studentId` (query, optional): Filter submissions by a specific student ID.
- `status` (query, optional): Filter submissions by status (e.g., submitted, pending, graded).
- `page` (query, optional): Specify the page of results to retrieve for pagination.
- `limit` (query, optional): Limit the number of results per page for pagination.

Example Code

```python
import requests

def get_student_submissions(course_id, post_id, student_id):
    """
    Fetch student submissions for a post with attachments in a course.

    Parameters:
    course_id (str): The ID of the course.
    post_id (str): The ID of the post.
    student_id (str): The ID of the student.

    Returns:
    dict: JSON response containing the student submissions.
    """
```

```
    # Define the API endpoint
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments/studentSubmissions/{stude
nt_id}"

    # Define the headers for the request
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',  # Replace with your actual access token
        'Accept': 'application/json'
    }

    try:
        # Make the API request
        response = requests.get(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the JSON response
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")
    except requests.exceptions.RequestException as err:
        print(f"An error occurred: {err}")

    return None

# Example usage
course_id = "1234567890"
post_id = "0987654321"
student_id = "1122334455"
submissions = get_student_submissions(course_id, post_id, student_id)
if submissions:
    print(submissions)
```

This code defines a function `get_student_submissions` that takes `course_id`, `post_id`, and `student_id` as parameters, constructs the appropriate URL for the API endpoint, sets up the necessary headers (including an authorization token), and makes a GET request to fetch the student submissions. It includes error handling for various potential exceptions and returns the JSON response if the request is successful.

Google API Documentation

Endpoint: courses.posts.addOnAttachments.studentSubmissions.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/posts/{postId}/addOnAttachments/{attachmentId}/studentSubmissions/{submissionId}

Description: Updates data associated with an add-on attachment submission. Requires the add-on to have been the original creator of the attachment and the attachment to have a positive `max_points` value set. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description:

Endpoint: PATCH classroom.courses.posts.addOnAttachments.studentSubmissions.patch

Common Use Cases:
- Updating the status of a student's submission attachment (e.g., marking it as approved or rejected).
- Modifying metadata associated with a student's submission attachment (e.g., changing the title or description).

Example Request:
```
PATCH /classroom.courses/{courseId}/posts/{postId}/studentSubmissions/{submissionId}/attachments/{attachmentId}
Content-Type: application/json

{
  "status": "approved",
  "description": "Great job on the assignment!"
}
```

Common Parameters:
- **courseId (Path Parameter):** The unique identifier for the course.
- **postId (Path Parameter):** The unique identifier for the post within the course.
- **submissionId (Path Parameter):** The unique identifier for the student's submission.
- **attachmentId (Path Parameter):** The unique identifier for the attachment within the submission.
- **status (Request Body):** The new status for the attachment (e.g., "approved", "rejected").
- **description (Request Body, optional):** Additional comments or notes regarding the status change.

Example Code

```python
import requests

def patch_student_submission(assignment_id, file_id, comment, token):
    """
    Patches the student submission with the given assignment_id and file_id.
    Adds a comment to the attachment.
```

Google API Documentation

```
    Parameters:
    assignment_id (str): The ID of the assignment.
    file_id (str): The ID of the file to be attached.
    comment (str): The comment to be added to the attachment.
    token (str): The authentication token for the API request.
    """
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{assignment_id}/studentSubmissions/{file_id}/attachments"
    headers = {
        'Authorization': f'Bearer {token}',
        'Content-Type': 'application/json'
    }
    data = {
        "comment": comment
    }

    try:
        response = requests.patch(url, headers=headers, json=data)
        response.raise_for_status()  # Will raise HTTPError for bad responses (4xx and 5xx)
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
assignment_id = "1234567890"
file_id = "0987654321"
comment = "This is a comment for the file attachment."
token = "YOUR_AUTH_TOKEN_HERE"

result = patch_student_submission(assignment_id, file_id, comment, token)
if result:
    print("Patch successful:", result)
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWorkMaterials/{id}

Description: Returns a course work material. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work material, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or course work material does not exist.

AI-Generated Documentation

GET classroom.courses.courseWorkMaterials.get

This endpoint retrieves the coursework materials for a specific course in a classroom.

Common Use Cases:
- Fetching coursework materials for review.
- Downloading course documents.
- Integrating classroom materials into a learning management system.

Example Request:
```
GET https://www.googleapis.com/classroom/v1/courses/{courseId}/courseWorkMaterials?key={YOUR_API_KEY}
```

Common Parameters:
- **courseId**: (Required) The unique identifier for the course.
- **key**: (Optional) The API key for authentication.

```json
{
  "courseId": "1234567890",
  "key": "YOUR_API_KEY"
}
```

Example Code

```python
import requests

def get_course_work_materials(course_id, material_id):
    """
    Function to fetch course work materials from the API.

    :param course_id: The ID of the course.
    :param material_id: The ID of the course material to fetch.
```

```
    :return: Response data if successful, or error message if not.
    """

    # API endpoint URL
            url   =   "https://classroom.googleapis.com/v1/courses/{}/courseWork/{}/materials".format(course_id,
material_id)

    # Headers for the request
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',  # Replace with a valid access token
        'Accept': 'application/json'
    }

    try:
        # Send GET request to the API
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            return response.json()  # Return the JSON response data
        else:
            return "Error: Unable to fetch course work materials. Status code: {}".format(response.status_code)

    except requests.exceptions.RequestException as e:
        # Handle any errors that occur during the request
        return "Error: An exception occurred - {}".format(e)

# Example usage:
course_id = '1234567890'
material_id = 'abcdefghij'

materials = get_course_work_materials(course_id, material_id)
print(materials)
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.getAddOnContext

HTTP Method: GET

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnContext

Description: Gets metadata for Classroom add-ons in the context of a specific post. To maintain the integrity of its own data and permissions model, an add-on should call this to validate query parameters and the requesting user's role whenever the add-on is opened in an [iframe](https://developers.google.com/classroom/add-ons/get-started/iframes/iframes-overview). This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint: GET classroom.courses.courseWorkMaterials.getAddOnContext

Common Use Cases:
- Retrieve context information for course work materials within a specific course.
- Integrate with educational add-ons to display relevant course materials.
- Facilitate the contextual display of assignments, resources, and other course materials.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWorkMaterials/{courseWorkMaterialId}/getAddOnContext
```

Common Parameters:
- **courseId (string):** The unique identifier for the course.
- **courseWorkMaterialId (string):** The unique identifier for the course work material.
- **alt (string, optional):** Data format for the response. Default is JSON.
- **fields (string, optional):** Selector specifying which fields to include in a partial response.
- **key (string, optional)**: API key. Required unless you provide an OAuth 2.0 token.

Example Code

```python
import requests

# Define the API endpoint
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/addOnContext"

# Define the parameters for the API request
params = {
    "courseId": "1234567890",  # Replace with a valid course ID
    "courseWorkId": "1234567890"  # Replace with a valid course work ID
}
```

```
# Define the headers for the API request
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with a valid access token
}

try:
    # Make the API request
        response  =  requests.get(url.format(courseId=params['courseId'],  courseWorkId=params['courseWorkId']),
headers=headers)

    # Check if the request was successful
    response.raise_for_status()

    # Parse the JSON response
    data = response.json()

    # Print the response data
    print(data)

except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as req_err:
    print(f"Error occurred: {req_err}")
except ValueError as json_err:
    print(f"Error parsing JSON: {json_err}")
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWorkMaterials/{id}

Description: Updates one or more fields of a course work material. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `FAILED_PRECONDITION` if the requested course work material has already been deleted. * `NOT_FOUND` if the requested course or course work material does not exist

AI-Generated Documentation

Endpoint: PATCH classroom.courses.courseWorkMaterials.patch

Common Use Cases:
- Update the properties of course work materials, such as titles, descriptions, or attachments.
- Modify the visibility or availability of course work materials for students.
- Correct errors or make minor updates to course materials without deleting and recreating them.

Example Request:

```http
PATCH https://api.example.com/classroom/courses/courseId/courseWorkMaterials/materialId
Content-Type: application/json

{
  "title": "Updated Course Material Title",
  "description": "Updated description for the course material.",
  "visibility": "public"
}
```

Common Parameters:

- **courseId**: Identifier for the course to which the course work material belongs.
- **materialId**: Identifier for the specific course work material to be updated.
- **title**: Optional. The new title for the course work material.
- **description**: Optional. The new description for the course work material.
- **visibility**: Optional. The visibility setting for the course work material (e.g., "public", "private").
- **attachments**: Optional. Any new attachments or updates to existing attachments for the course work material.

Example Code

```python
import requests

def patch_course_work_materials(course_id, course_work_id, material_id, new_title, new_description):
    # Define the API endpoint URL
```

```
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/materials/{material_id}"

    # Define the headers including authorization
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',  # Replace with your actual access token
        'Content-Type': 'application/json'
    }

    # Define the data to be sent in the PATCH request
    data = {
        'title': new_title,
        'description': new_description
    }

    try:
        # Make the PATCH request
        response = requests.patch(url, headers=headers, json=data)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the response JSON
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # Print HTTP errors
    except Exception as err:
        print(f'Other error occurred: {err}')  # Print other errors

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
material_id = '123abc456def'
new_title = 'Updated Material Title'
new_description = 'This is the updated description of the material.'

patch_course_work_materials(course_id, course_work_id, material_id, new_title, new_description)
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.create

HTTP Method: POST

Path: v1/courses/{courseId}/courseWorkMaterials

Description: Creates a course work material. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course, create course work material in the requested course, share a Drive attachment, or for access errors. * `INVALID_ARGUMENT` if the request is malformed or if more than 20 * materials are provided. * `NOT_FOUND` if the requested course does not exist. * `FAILED_PRECONDITION` for the following request error: * AttachmentNotVisible

AI-Generated Documentation

Common Use Cases:
- Teachers can upload course materials such as documents, videos, or links for students to access.
- Administrators can add supplementary materials to specific courses.
- Students can submit assignments or projects that will be reviewed by the teacher.

Example Request:
```json
{
  "courseId": "12345",
  "materialName": "Introduction to Algorithms",
  "materialType": "Document",
  "materialUrl": "http://example.com/intro_alg.pdf"
}
```

Common Parameters:
- **courseId** (string): The unique identifier for the course to which the material will be added.
- **materialName** (string): The name or title of the material being uploaded.
- **materialType** (string): The type of material (e.g., Document, Video, Link).
- **materialUrl** (string): The URL or file path where the material is hosted.

Example Code

```python
import requests

# Define the API endpoint URL
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWorkMaterials"

# Define the headers for authentication
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"
}

# Define the payload with realistic parameter names and values
```

```
payload = {
    "title": "Introduction to Python",
    "description": "Material for the first Python class",
    "materials": [
        {
            "link": "https://example.com/python-intro.pdf",
            "type": "link"
        }
    ]
}

# Define the course ID
course_id = "1234567890"

# Replace the placeholder in the URL with the actual course ID
url = url.replace("{courseId}", course_id)

# Make the API request
response = requests.post(url, headers=headers, json=payload)

# Check for HTTP errors
if response.status_code == 200:
    # Successful response
    print("Course material created successfully")
    print(response.json())
else:
    # Handle errors
    print(f"Failed to create course material. Status code: {response.status_code}")
    print(f"Error message: {response.text}")
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWorkMaterials

Description: Returns a list of course work material that the requester is permitted to view. Course students may only view `PUBLISHED` course work material. Course teachers and domain administrators may view all course work material. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

Description

Common Use Cases:
- Retrieve a list of course materials for a specific course.
- Check the availability and details of coursework materials.
- Integrate course materials into a learning management system.

Example Request:
```http
GET /v1/classroom.courses.courseWorkMaterials.list?courseId=COURSE_ID&materialType=TYPE
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **materialType**: The type of material to retrieve (e.g., "homework", "assignment", "quiz").
- **pageToken**: Token for pagination, used to retrieve the next set of results.
- **maxResults**: Maximum number of results to return per page.

Google API Documentation

Endpoint: courses.courseWorkMaterials.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/courseWorkMaterials/{id}

Description: Deletes a course work material. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work material item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding course work material, if the requesting user is not permitted to delete the requested course or for access errors. * `FAILED_PRECONDITION` if the requested course work material has already been deleted. * `NOT_FOUND` if no course exists with the requested ID.

AI-Generated Documentation

Technical Description for DELETE classroom.courses.courseWorkMaterials.delete

Endpoint: DELETE classroom.courses.courseWorkMaterials.delete

Common Use Cases:
- Removing a specific coursework material that is no longer relevant.
- Cleaning up obsolete or outdated materials from a course.
- Managing course content to ensure only current and necessary materials are available.

Example Request:
```http
DELETE /v1/classroom/courses/{courseId}/courseWorkMaterials/{materialId}
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:

- **courseId (Path Parameter):** The unique identifier for the course from which the material will be deleted.
- **materialId (Path Parameter):** The unique identifier for the coursework material to be deleted.
- **Authorization (Header):** A Bearer token used for authentication and authorization purposes.

Example Code

```python
import requests

def delete_course_work_material(course_id, assignment_id, material_id, api_key):
    # Define the API endpoint URL
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{assignment_id}/materials/{material_id}"

    # Define the headers including the API key for authorization
    headers = {
```

```
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }

    # Send the DELETE request to the API endpoint
    response = requests.delete(url, headers=headers)

    # Check if the request was successful
    if response.status_code == 204:
        print("Course work material deleted successfully.")
    else:
        # Handle errors by printing the status code and error message
        print(f"Error deleting course work material: {response.status_code}")
        try:
            error_data = response.json()
            print(f"Error details: {error_data.get('error', 'Unknown error')}")
        except ValueError:
            print(f"Error details: {response.text}")

# Example usage
course_id = "1234567890"
assignment_id = "9876543210"
material_id = "1010101010"
api_key = "YOUR_API_KEY"

delete_course_work_material(course_id, assignment_id, material_id, api_key)
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.addOnAttachments.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnAttachments/{attachmentId}

Description: Updates an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Endpoint: PATCH classroom.courses.courseWorkMaterials.addOnAttachments.patch

Common Use Cases:
- Update the metadata of an attachment already added to coursework materials.
- Modify the visibility or access permissions of an attachment.
- Correct errors in attachment details without removing the attachment itself.

Example Request:
```
PATCH https://api.example.com/v1/classroom.courses.courseWorkMaterials/addOnAttachments/{attachmentId}.patch
Content-Type: application/json

{
  "attachmentId": "1234567890",
  "title": "Updated Project Proposal",
  "description": "This is the updated project proposal document.",
  "visibility": "visibleToAll"
}
```

Common Parameters:
- `attachmentId (Path Parameter)`: The unique identifier of the attachment to be updated.
- `title (Optional)`: The new title for the attachment.
- `description (Optional)`: The new description for the attachment.
- `visibility (Optional)`: The new visibility setting for the attachment (e.g., "visibleToAll", "visibleToStudents", "visibleToTeachers").
- `accessPermissions (Optional)`: The new access permissions for the attachment (e.g., "view", "edit", "comment").

Example Code

```python
import requests

# Define the API endpoint
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/attachments"
```

```
# Define the headers for the request, including the authentication token
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the data to be sent in the PATCH request
data = {
    "title": "Updated Title",  # New title for the attachment
    "link": "https://example.com/updated-link"  # Updated link for the attachment
}

# Define the parameters for the URL
params = {
    "courseId": "course123",  # Replace with a valid course ID
    "courseWorkId": "courseWork456",  # Replace with a valid course work ID
    "materialId": "material789"  # Replace with a valid material ID
}

# Construct the full URL with parameters
full_url = url.format(**params)

try:
    # Make the PATCH request
    response = requests.patch(full_url, headers=headers, json=data)

    # Check if the request was successful
    if response.status_code == 200:
        print("Attachment updated successfully.")
        print(response.json())
    else:
        # Handle errors
        print(f"Failed to update attachment. Status code: {response.status_code}")
        print(response.json())
except requests.exceptions.RequestException as e:
    # Handle any request exceptions
    print(f"An error occurred: {e}")
```

This code demonstrates how to use the `requests` library to send a PATCH request to the `classroom.courses.courseWorkMaterials.addOnAttachments.patch` endpoint. It includes realistic parameter names and values, proper error handling, and inline comments for clarity.

Google API Documentation

Endpoint: courses.courseWorkMaterials.addOnAttachments.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnAttachments/{attachmentId}

Description: Returns an add-on attachment. Requires the add-on requesting the attachment to be the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.courseWorkMaterials.addOnAttachments.get

Common Use Cases:
- Retrieve attachments added to course work materials.
- Fetch specific attachments for a particular course work material.
- Gather metadata and URLs of attachments for further processing.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/addOnAttachments?attachmentsId={attachmentId}
```

Common Parameters:
- **courseId** (Path Parameter, required): The unique identifier for the course.
- **courseWorkId** (Path Parameter, required): The unique identifier for the course work.
- **materialId** (Path Parameter, required): The unique identifier for the material within the course work.
- **attachmentsId** (Query Parameter, optional): The unique identifier for the attachment to retrieve.
- **fields** (Query Parameter, optional): Selector specifying which fields to include in a partial response.
- **quotaUser** (Query Parameter, optional): An opaque string that represents a user for quota purposes.

Example Code

```python
import requests

# Define the API endpoint and the required parameters
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/attachments/{attachmentId}"

# Replace these with actual values
course_id = "1234567890"
course_work_id = "courseWork_123"
material_id = "material_456"
attachment_id = "attachment_789"
```

```
access_token = "YOUR_ACCESS_TOKEN"

# Prepare the full URL
full_url = url.format(courseId=course_id, courseWork=course_work_id, materialId=material_id,
attachmentId=attachment_id)

# Set up the headers including the authorization token
headers = {
    "Authorization": f"Bearer {access_token}"
}

try:
    # Make the GET request to the API
    response = requests.get(full_url, headers=headers)

    # Raise an exception for HTTP errors
    response.raise_for_status()

    # Parse the JSON response
    data = response.json()

    # Print the response data
    print(data)

except requests.exceptions.HTTPError as http_err:
    # Handle HTTP errors
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as req_err:
    # Handle general request errors
    print(f"Request error occurred: {req_err}")
except ValueError as val_err:
    # Handle JSON decoding errors
    print(f"Error decoding JSON: {val_err}")
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.addOnAttachments.create

HTTP Method: POST

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnAttachments

Description: Creates an add-on attachment under a post. Requires the add-on to have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for POST classroom.courses.courseWorkMaterials.addOnAttachments.create

Common Use Cases:
- Adding attachments to course materials for students to access.
- Providing additional resources or supplementary materials for a course.
- Uploading documents, images, or other files to enhance the learning experience.

Example Request:
```json
{
  "courseId": "12345",
  "materialId": "67890",
  "file": {
    "name": "example_document.pdf",
    "data": "base64_encoded_string"
  }
}
```

Common Parameters:
- **courseId**: (string) The unique identifier for the course.
- **materialId**: (string) The unique identifier for the course material.
- **file**: (object) The file to be attached.
  - **name**: (string) The name of the file.
  - **data**: (string) The base64-encoded data of the file.

Example Code

```python
import requests

def add_on_attachments_to_course_work_materials(course_id, course_work_id, attachment_title, attachment_file_id):
    """
    Function to add attachments to course work materials in Google Classroom.
    """
    # Define the API endpoint URL
```

```
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/materials"

    # Create the request payload with attachment details
    payload = {
        "title": attachment_title,
        "attachmentId": attachment_file_id
    }

    # Define headers, including the authorization header with OAuth 2.0 token
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }

    try:
        # Make the API request to add attachments
        response = requests.post(url, json=payload, headers=headers)

        # Check if the request was successful
        response.raise_for_status()

        # Return the response JSON
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = "1234567890"
course_work_id = "12345"
attachment_title = "Sample Attachment"
attachment_file_id = "65432109876543210"

result = add_on_attachments_to_course_work_materials(course_id, course_work_id, attachment_title,
attachment_file_id)
if result:
    print("Attachment added successfully:", result)
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.addOnAttachments.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnAttachments/{attachmentId}

Description: Deletes an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for DELETE classroom.courses.courseWorkMaterials.addOnAttachments.delete

Common Use Cases:
- Delete specific attachments from coursework materials.
- Remove outdated or incorrect files associated with a particular course assignment.
- Clean up unused attachments to optimize storage space.

Example Request:
```http
DELETE /classroom/v1/courses/{courseId}/courseWorkMaterials/{materialId}/addOnAttachments/{attachmentId}
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId** (Path Parameter): Unique identifier for the course.
- **materialId** (Path Parameter): Unique identifier for the coursework material.
- **attachmentId** (Path Parameter): Unique identifier for the attachment to be deleted.
- **Authorization** (Header): Bearer token for authenticating the API request.

Example Code

```python
import requests
from requests.exceptions import HTTPError

# Define the endpoint URL and headers
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/attachments/{attachmentId}"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the parameters
course_id = "your_course_id"
course_work_id = "your_course_work_id"
```

```
material_id = "your_material_id"
attachment_id = "your_attachment_id"

# Construct the full URL with the parameters
full_url = url.format(
    courseId=course_id,
    courseWorkId=course_work_id,
    materialId=material_id,
    attachmentId=attachment_id
)

# Make the DELETE request
try:
    response = requests.delete(full_url, headers=headers)
    response.raise_for_status()  # Raise an exception for HTTP errors
    print("Attachment deleted successfully.")
except HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except Exception as err:
    print(f"An error occurred: {err}")
```

Google API Documentation

Endpoint: courses.courseWorkMaterials.addOnAttachments.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWorkMaterials/{itemId}/addOnAttachments

Description: Returns all attachments created by an add-on under the post. Requires the add-on to have active attachments on the post or have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.courseWorkMaterials.addOnAttachments.list

Common Use Cases:
- Retrieving a list of attachments associated with specific coursework materials.
- Verifying the availability and status of attachments for a given coursework item.
- Gathering metadata related to attachments for further processing or display.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialsId}/attachments
Authorization: Bearer {access_token}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `courseWorkId` (string): The unique identifier for the coursework.
- `materialsId` (string): The unique identifier for the coursework materials.
- `pageToken` (string, optional): Token used to access the next page of results. This is used for pagination.
- `maxResults` (integer, optional): Maximum number of results to return. Default is 100.

```json
{
  "courseId": "1234567890",
  "courseWorkId": "abcdefghij",
  "materialsId": "klmnopqrst"
}
```

Example Code

```python
import requests

def list_course_work_materials_attachments(course_id, course_work_id, access_token):
```

```
    # Define the API endpoint and headers
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWorks/{course_work_id}/materials:list"
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }

    # Define the query parameters
    params = {
        'pageSize': 10,  # Maximum number of results to return
        'pageToken': None  # Token for pagination, can be None for the first request
    }

    try:
        # Make the API request
        response = requests.get(url, headers=headers, params=params)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Print or return the data as needed
        print(data)

        # Check for pagination token and print it if available
        next_page_token = data.get('nextPageToken')
        if next_page_token:
            print(f"Next page token: {next_page_token}")

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Request error occurred: {req_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = "1234567890"
course_work_id = "0987654321"
access_token = "your_access_token_here"

list_course_work_materials_attachments(course_id, course_work_id, access_token)
```

Google API Documentation

Endpoint: courses.courseWork.create

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork

Description: Creates course work. The resulting course work (and corresponding student submissions) are associated with the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to make the request. Classroom API requests to modify course work and student submissions must be made with an OAuth client ID from the associated Developer Console project. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course, create course work in the requested course, share a Drive attachment, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist. * `FAILED_PRECONDITION` for the following request error: * AttachmentNotVisible

AI-Generated Documentation

Technical Description for POST classroom.courses.courseWork.create

Common Use Cases
- Creating new assignments or homework for a specific course.
- Automating the generation of coursework from a learning management system.
- Allowing students to submit projects or papers.

Example Request
```json
{
  "courseId": "course1234",
  "title": "Math Homework",
  "description": "Complete the exercises on pages 50-55.",
  "dueDate": "2023-12-31T23:59:59Z",
  "materials": ["textbook", "online_quiz"],
  "maxPoints": 100
}
```

Common Parameters
- **courseId**: The unique identifier for the course.
- **title**: The title of the coursework.
- **description**: A detailed description of the coursework.
- **dueDate**: The deadline for the coursework (ISO 8601 format).
- **materials**: An optional array of materials required for the coursework.
- **maxPoints**: The maximum points the coursework is worth.

Example Code

```python
import requests
```

```
# Define the API endpoint and parameters
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork"
course_id = "1234567890"  # Replace with a real course ID
title = "New Assignment"
description = "This is the description of the new assignment."
materials = [
    {
        "link": {
            "url": "https://example.com/material1"
        }
    },
    {
        "link": {
            "url": "https://example.com/material2"
        }
    }
]
due_date = {
    "year": 2023,
    "month": 10,
    "day": 15
}
max_points = 100

# Define the payload
payload = {
    "title": title,
    "description": description,
    "materials": materials,
    "dueDate": due_date,
    "workType": "ASSIGNMENT",
    "maxPoints": max_points
}

# Define headers
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",  # Replace with a real access token
    "Content-Type": "application/json"
}

# Make the API request
response = requests.post(url.format(courseId=course_id), json=payload, headers=headers)

# Check for successful response
if response.status_code == 200:
    print("Assignment created successfully:", response.json())
else:
    # Print the error message if the request failed
    print("Failed to create assignment:", response.status_code, response.text)
```

```

This script creates a new assignment in a Google Classroom course using the `classroom.courses.courseWork.create` endpoint. It includes realistic parameter names and values, proper error handling, and inline explanations as comments.
```

Google API Documentation

Endpoint: courses.courseWork.getAddOnContext

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnContext

Description: Gets metadata for Classroom add-ons in the context of a specific post. To maintain the integrity of its own data and permissions model, an add-on should call this to validate query parameters and the requesting user's role whenever the add-on is opened in an [iframe](https://developers.google.com/classroom/add-ons/get-started/iframes/iframes-overview). This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint Description: GET classroom.courses.courseWork.getAddOnContext

Common Use Cases:
- Retrieve context information for add-ons within a coursework.
- Enable add-ons to provide tailored functionalities based on coursework details.
- Integrate third-party applications with classroom coursework.

Example Request:
```
GET https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/getAddOnContext
```

Common Parameters:
- **courseId** (required): The unique identifier for the course.
- **courseWorkId** (required): The unique identifier for the coursework within the specified course.

The request URL should be constructed with these placeholder values replaced by actual data.

Example Code

```python
import requests

def get_add_on_context(api_key, course_id, course_work_id):
    """
    Fetch the add-on context for a specific course work in a classroom.

    :param api_key: The API key for authentication.
    :param course_id: The ID of the course.
    :param course_work_id: The ID of the course work.
    :return: The JSON response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/addOnContext"
    headers = {
```

```
        'Authorization': f'Bearer {api_key}',
        'Content-Type': 'application/json'
    }

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an error for bad status codes
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # Print HTTP errors
    except requests.exceptions.RequestException as req_err:
        print(f'Error occurred: {req_err}')  # Print general request errors
    except ValueError as json_err:
        print(f'JSON decode error: {json_err}')  # Print JSON decode errors

# Example usage
api_key = 'your_api_key_here'
course_id = '1234567890'
course_work_id = '0987654321'

add_on_context = get_add_on_context(api_key, course_id, course_work_id)
if add_on_context:
    print(add_on_context)
```

This code defines a function `get_add_on_context` that fetches the add-on context for a specific course work in a Google Classroom. It includes proper error handling for HTTP errors, request exceptions, and JSON decoding errors. The example usage at the end demonstrates how to call the function with realistic parameter values.

Google API Documentation

Endpoint: courses.courseWork.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{id}

Description: Updates one or more fields of a course work. See google.classroom.v1.CourseWork for details of which fields may be updated and who may change them. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding course work, if the user is not permitted to make the requested modification to the student submission, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `FAILED_PRECONDITION` if the requested course work has already been deleted. * `NOT_FOUND` if the requested course or course work does not exist.

AI-Generated Documentation

Technical Description for `classroom.courses.courseWork.patch`

Common Use Cases:
- Update a coursework item's details such as title, description, or due date.
- Modify the grading parameters or criteria for an assignment or project.
- Adjust coursework visibility or accessibility settings.

Example Request:
```
PATCH https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{id}
Content-Type: application/json
Authorization: Bearer {access_token}

{
  "title": "Updated Assignment Title",
  "description": "Updated description of the assignment.",
  "dueDate": {
    "year": 2023,
    "month": 12,
    "day": 31
  },
  "workType": "ASSIGNMENT",
  "state": "PUBLISHED"
}
```

Common Parameters:

- **courseId**: (string, required) The identifier of the course.
- **id**: (string, required) The identifier of the coursework item.
- **title**: (string) The title of the coursework item.
- **description**: (string) A detailed description of the coursework item.

- **dueDate**: (object) The due date for the coursework.
  - `year`: (integer) The year of the due date.
  - `month`: (integer) The month of the due date.
  - `day`: (integer) The day of the due date.
- **workType**: (string) The type of coursework (e.g., "ASSIGNMENT", "HOMEWORK").
- **state**: (string) The state of the coursework (e.g., "DRAFT", "PUBLISHED").

Example Code

```python
import requests

# Define the API endpoint and headers
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the payload with realistic parameter names and values
payload = {
    "title": "Updated Assignment Title",
    "description": "This is the updated description of the assignment.",
    "state": "PUBLISHED",
    "workType": "HOMEWORK_ASSIGNMENT"
}

# Include proper error handling
try:
    # Make the PATCH request
    response = requests.patch(url, headers=headers, json=payload)
    response.raise_for_status()  # Raise an exception for HTTP errors
    # Return the updated course work
    updated_course_work = response.json()
    print("Course work updated successfully:", updated_course_work)
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as req_err:
    print(f"Error occurred: {req_err}")
```

This code snippet demonstrates how to use the `requests` library in Python to make a PATCH request to the `classroom.courses.courseWork.patch` endpoint. It includes realistic parameter names and values, proper error handling, and prints the updated course work if the request is successful.

Google API Documentation

Endpoint: courses.courseWork.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork

Description: Returns a list of course work that the requester is permitted to view. Course students may only view `PUBLISHED` course work. Course teachers and domain administrators may view all course work. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

API Endpoint: GET classroom.courses.courseWork.list

Common Use Cases:
- Retrieve a list of coursework assignments for a specific course.
- Monitor the status and due dates of assignments.
- Integrate coursework data into educational management systems.

Example Request:
```
GET https://classroom.googleapis.com/v1/courses/{courseId}/courseWork
Authorization: Bearer {access_token}
```

Common Parameters:
- **courseId**: The ID of the course. This is a required parameter.
- **pageSize**: The maximum number of coursework items to return. If not specified, the server will determine a default value.
- **pageToken**: The nextPageToken value returned from a previous request, used for pagination.
- **orderBy**: The sorting order of the coursework items. Acceptable values include "title" and "dueDate".
- **filter**: A filter query string for filtering the coursework items by certain criteria, such as due date or status.

Example Code

```python
import requests

def list_course_work(service_account_credentials, course_id):
    """
    Lists the coursework for a given course.

    Args:
    service_account_credentials (str): The service account credentials JSON string.
    course_id (str): The ID of the course to list coursework for.

    Returns:
```

```
    dict: The JSON response from the API.
    """

    # Define the API endpoint URL
    url = "https://classroom.googleapis.com/v1/courses/" + course_id + "/courseWork"

    # Set up the headers with the service account credentials
    headers = {
        "Authorization": f"Bearer {service_account_credentials}"
    }

    # Set up the parameters for the request
    params = {
        "pageSize": 10,  # Number of coursework items to return per page
        "orderBy": "updateTime desc"  # Sort by update time in descending order
    }

    try:
        # Send the GET request to the API
        response = requests.get(url, headers=headers, params=params)

        # Check if the request was successful
        response.raise_for_status()

        # Return the JSON response
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"Other error occurred: {err}")
    return None

# Example usage
service_account_credentials = "your_service_account_credentials_here"
course_id = "1234567890"

course_work = list_course_work(service_account_credentials, course_id)
if course_work:
    print(course_work)
```

This Python code defines a function `list_course_work` that makes a GET request to the Google Classroom API to list coursework for a given course. It includes proper error handling using try-except blocks and returns the JSON response from the API if the request is successful. The `headers` and `params` are set up to include the necessary authorization and query parameters, respectively. The example usage at the end demonstrates how to call the function with realistic parameter values.

Google API Documentation

Endpoint: courses.courseWork.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{id}

Description: Returns course work. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or course work does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.courseWork.get

Common Use Cases:
- Retrieve a specific piece of coursework for a given course.
- Access detailed information about assignments or projects.
- Obtain student submissions for a particular coursework.
- Verify the status and deadlines of coursework assignments.

Example Request:
```
GET https://www.googleapis.com/classroom/v1/courses/{courseId}/courseWork/{courseWorkId}?key={YOUR_API_KEY}
```

Common Parameters:
- **courseId** (required): The ID of the course for which the coursework is to be retrieved.
- **courseWorkId** (required): The ID of the specific coursework within the course.
- **key** (required): Your API key for authentication.
- **projection** (optional): Determines the fields to be included in the response. Possible values: `full`, `basic`.
- **fields** (optional): Selector specifying which fields to include in a partial response.

Example Code

```python
import requests

def get_course_work(course_id, course_work_id, api_key):
    # Define the API endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}"

    # Set up the headers with the API key
    headers = {
        'Authorization': f'Bearer {api_key}'
    }

    # Make the API request
    response = requests.get(url, headers=headers)
```

```
    # Check if the request was successful
    if response.status_code == 200:
        # Return the JSON response
        return response.json()
    else:
        # Print the error message
        print(f"Error: {response.status_code}")
        print(response.json())
        return None

# Example usage
api_key = 'YOUR_API_KEY'
course_id = '123456'
course_work_id = '654321'

course_work_data = get_course_work(course_id, course_work_id, api_key)

if course_work_data:
    print("Course Work Data:", course_work_data)
else:
    print("Failed to retrieve course work data.")
```

Google API Documentation

Endpoint: courses.courseWork.modifyAssignees

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{id}:modifyAssignees

Description: Modifies assignee mode and options of a coursework. Only a teacher of the course that contains the coursework may call this method. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or course work does not exist.

AI-Generated Documentation

Technical Description for POST classroom.courses.courseWork.modifyAssignees

Common Use Cases:
- Modifying the list of students assigned to a specific course work.
- Adding or removing individual students from a course work assignment.
- Ensuring that the correct students are assigned to a particular task or assignment.

Example Request:
```http
POST https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/modifyAssignees
Content-Type: application/json
Authorization: Bearer {access_token}

{
  "addStudentIds": ["studentId1", "studentId2"],
  "removeStudentIds": ["studentId3"]
}
```

Common Parameters:
- **addStudentIds**: (Optional) A list of student IDs to add to the course work assignees.
- **removeStudentIds**: (Optional) A list of student IDs to remove from the course work assignees.
- **courseId**: (Path Parameter) The ID of the course.
- **courseWorkId**: (Path Parameter) The ID of the course work.

Example Code

```python
import requests

def modify_coursework_assignees(course_id, coursework_id, assignees):
    """
    Modify the assignees of a coursework assignment.

    Args:
```

```
    course_id (str): The ID of the course.
    coursework_id (str): The ID of the coursework.
    assignees (list): A list of user IDs to assign to the coursework.

    Returns:
    dict: The response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{coursework_id}/modifyAssignees"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }
    data = {
        "assignees": assignees
    }

    try:
        response = requests.post(url, headers=headers, json=data)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

# Example usage
course_id = "1234567890"
coursework_id = "9876543210"
assignees = ["user1@example.com", "user2@example.com"]

response = modify_coursework_assignees(course_id, coursework_id, assignees)
if response:
    print("Assignees modified successfully:", response)
```

Endpoint: courses.courseWork.updateRubric

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubric

Description: Updates a rubric. See google.classroom.v1.Rubric for details of which fields can be updated. Rubric update capabilities are [limited](/classroom/rubrics/limitations) once grading has started. The requesting user and course owner must have rubrics creation capabilities. For details, see [licensing requirements](https://developers.google.com/classroom/rubrics/limitations#license-requirements). This request must be made by the Google Cloud console of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the parent course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project didn't create the corresponding course work, if the user isn't permitted to make the requested modification to the rubric, or for access errors. This error code is also returned if grading has already started on the rubric. * `INVALID_ARGUMENT` if the request is malformed and for the following request error: * `RubricCriteriaInvalidFormat` * `NOT_FOUND` if the requested course, course work, or rubric doesn't exist or if the user doesn't have access to the corresponding course work. * `INTERNAL` if grading has already started on the rubric.

AI-Generated Documentation

API Endpoint: PATCH classroom.courses.courseWork.updateRubric

Common Use Cases:
- Update the rubric criteria for a coursework assignment.
- Modify scoring metrics for a coursework rubric.
- Adjust rubric guidelines based on changes in course requirements.

Example Request:
```
PATCH https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/updateRubric
Content-Type: application/json

{
  "rubric": {
    "title": "New Rubric Title",
    "points": "100",
    "criterion": [
      {
        "title": "Criteria 1",
        "maxPoints": 20,
        "description": "Description of Criteria 1"
      },
      // Additional criteria can be added here
    ]
  }
}
```

Common Parameters:

- **courseId** (Path, Required): The ID of the course.
- **courseWorkId** (Path, Required): The ID of the coursework.
- **updateRubricRequest** (Body, Required):
  - **title** (String, Optional): The title of the rubric.
  - **points** (Integer, Optional): The total points for the rubric.
  - **criterion** (Array, Optional): An array of criteria, each with:
    - **title** (String, Required): The title of the criterion.
    - **maxPoints** (Integer, Required): The maximum points for the criterion.
    - **description** (String, Optional): The description of the criterion.

Example Code

```python
import requests

# Define the API endpoint and the required headers
url = "https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{courseWork_id}/updateRubric"
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the parameters for the request
params = {
    "course_id": "1234567890",  # Replace with a real course ID
    "courseWork_id": "abcdefghijklmnopqrstuvwxyz"  # Replace with a real courseWork ID
}

# Define the data payload for the request
data = {
    "rubric": {
        "title": "New Rubric Title",
        "description": "This is a new rubric description.",
        "items": [
            {
                "title": "Item 1",
                "description": "Description for item 1.",
                "points": 10
            },
            {
                "title": "Item 2",
                "description": "Description for item 2.",
                "points": 5
            }
        ]
    }
}

# Make the API request
```

```
response = requests.put(url.format(**params), headers=headers, json=data)

# Check for errors in the response
if response.status_code == 200:
    print("Rubric updated successfully.")
else:
    print(f"Failed to update rubric: {response.status_code}")
    print(response.json())  # Print the error message from the API
```

Google API Documentation

Endpoint: courses.courseWork.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/courseWork/{id}

Description: Deletes a course work. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding course work, if the requesting user is not permitted to delete the requested course or for access errors. * `FAILED_PRECONDITION` if the requested course work has already been deleted. * `NOT_FOUND` if no course exists with the requested ID.

AI-Generated Documentation

API Endpoint: DELETE classroom.courses.courseWork.delete

Common Use Cases:
- Deleting a specific assignment or homework item from a course.
- Removing outdated or irrelevant coursework to clean up the course content.
- Managing course content dynamically based on student progress or feedback.

Example Request:
```http
DELETE /v1/courses/{courseId}/courseWork/{courseWorkId}
Headers:
  Authorization: Bearer {access_token}
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **courseWorkId**: The unique identifier for the specific coursework item to be deleted.
- **Authorization**: Bearer token for authentication.

Example Code

```python
import requests

# Define the endpoint URL and API key
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{id}"
api_key = "YOUR_API_KEY"
course_id = "1234567890"  # Replace with a realistic course ID
course_work_id = "0987654321"  # Replace with a realistic course work ID

# Define the headers for the API request
headers = {
    "Authorization": f"Bearer {api_key}"
}
```

```
# Define the parameters for the request
params = {
    "id": course_work_id
}

# Make the API request to delete the course work
response = requests.delete(url.format(courseId=course_id, id=course_work_id), headers=headers, params=params)

# Check the response status code
if response.status_code == 204:
    print("Course work deleted successfully.")
elif response.status_code == 404:
    print("Course work not found.")
elif response.status_code == 401:
    print("Unauthorized access. Check your API key.")
else:
    print(f"An error occurred: {response.status_code}")
    print("Error message:", response.json().get("error", {}).get("message", ""))
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.create

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments

Description: Creates an add-on attachment under a post. Requires the add-on to have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint Description

Endpoint: POST classroom.courses.courseWork.addOnAttachments.create

Common Use Cases
- Attach files, links, or other resources to a specific assignment in a course.
- Provide students with supplementary materials related to coursework.
- Enrich course content with additional resources that can support learning objectives.

Example Request
```json
{
  "courseWorkId": "12345",
  "attachments": [
    {
      "title": "Example Document",
      "fileId": "file123",
      "link": "https://example.com/document"
    },
    {
      "title": "Example Link",
      "link": "https://example.com/resource"
    }
  ]
}
```

Common Parameters
- **courseWorkId** (string): The unique identifier for the coursework to which the attachments will be added.
- **attachments** (array of objects): An array of attachment objects, each containing:
  - **title** (string): The title of the attachment.
  - **fileId** (optional, string): The unique identifier of the file to be attached.
  - **link** (optional, string): The URL of the resource to be linked.

Example Code

```python
```

```
import requests

def add_attachments_to_coursework(course_id, course_work_id, file_id, file_name, file_type):
    # Define the URL for the API endpoint
                                                                                    url                     =
"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/{course
_work_id}/attachments/create".format(
        course_id=course_id,
        course_work_id=course_work_id
    )

    # Define the headers for the request
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
        'Content-Type': 'application/json'
    }

    # Define the parameters for the request
    payload = {
        'file_id': file_id,
        'file_name': file_name,
        'file_type': file_type
    }

    try:
        # Make the POST request to add attachments to the coursework
        response = requests.post(url, headers=headers, json=payload)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the JSON response from the API
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except requests.exceptions.RequestException as err:
        print(f'Error occurred: {err}')
    except Exception as e:
        print(f'An unexpected error occurred: {e}')

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
file_id = 'file_id_example'
file_name = 'example_file.pdf'
file_type = 'application/pdf'

response = add_attachments_to_coursework(course_id, course_work_id, file_id, file_name, file_type)
```

```
print(response)
```
```
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments/{attachmentId}

Description: Deletes an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Description for API Endpoint: DELETE classroom.courses.courseWork.addOnAttachments.delete

Common Use Cases:
- Removing a specific attachment from a coursework assignment.
- Cleaning up outdated or irrelevant attachments from a coursework.
- Managing storage by deleting unused attachments.

Example Request:
```
DELETE /v1/classroom/courses/{courseId}/courseWork/{courseWorkId}/addOnAttachments/{attachmentId}
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **courseWorkId**: The unique identifier for the coursework assignment.
- **attachmentId**: The unique identifier for the attachment to be deleted.

Example Code

```python
import requests

# Define the API endpoint URL
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/addOnAttachments/{attachmentId}"

# Define the parameters for the API request, replace with realistic values
courseId = "1234567890"
courseWorkId = "9876543210"
attachmentId = "1234567890"

# Define the headers for authentication, replace 'your_access_token' with a valid access token
headers = {
    "Authorization": "Bearer your_access_token",
    "Content-Type": "application/json"
}
```

```
# Prepare the URL with the parameters
url = url.format(courseId=courseId, courseWorkId=courseWorkId, attachmentId=attachmentId)

# Make the DELETE request
response = requests.delete(url, headers=headers)

# Check for HTTP errors
if response.status_code == 204:
    print("Attachment deleted successfully.")
elif response.status_code == 404:
    print("Attachment not found.")
else:
    print(f"Failed to delete attachment. HTTP Status Code: {response.status_code}")
    print(f"Response: {response.text}")
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments

Description: Returns all attachments created by an add-on under the post. Requires the add-on to have active attachments on the post or have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Common Use Cases
- List all attachments added to a specific piece of coursework.
- Verify the attachments submitted by students for grading purposes.
- Generate a report of all attachments for a particular coursework item.

Example Request
```
GET https://api.example.com/classroom/courses/{courseId}/courseWork/{courseWorkId}/addOnAttachments/list
```

Common Parameters
- **courseId (path parameter)**: The unique identifier for the course.
- **courseWorkId (path parameter)**: The unique identifier for the specific coursework item.
- **pageToken (query parameter, optional)**: A token to retrieve the next set of results.
- **maxResults (query parameter, optional)**: The maximum number of results to return per page.

Example Code

```python
import google.auth
from googleapiclient.discovery import build
from googleapiclient.errors import HttpError

# Function to list attachments for a course work item
def list_attachments(course_id, course_work_id):
    try:
        # Authenticate and build the service
        credentials, project = google.auth.default()
        service = build('classroom', 'v1', credentials=credentials)

        # Define the request parameters
        request_params = {
            'courseId': course_id,
            'courseWorkId': course_work_id
        }
```

```
        # Make the API request
        results = service.courses().courseWork().studentSubmissions().list(**request_params).execute()

        # Process the results
        for student_submission in results.get('studentSubmissions', []):
            for attachment in student_submission.get('attachments', []):
                print(f"Attachment ID: {attachment['id']}, Attachment Title: {attachment['title']}")

    except HttpError as error:
        # Handle API request errors
        print(f'An error occurred: {error}')
        return None

    except Exception as e:
        # Handle other exceptions
        print(f'An unexpected error occurred: {e}')
        return None

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
list_attachments(course_id, course_work_id)
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments/{attachmentId}

Description: Returns an add-on attachment. Requires the add-on requesting the attachment to be the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint: GET classroom.courses.courseWork.addOnAttachments.get

Common Use Cases:
- Retrieving attachments added to a specific coursework item.
- Verifying the presence of attachments for a given coursework.
- Preparing for the downloading or viewing of coursework attachments.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseID}/courseWork/{courseWorkID}/addOnAttachments?key={API_KEY}
```

Common Parameters:
- **courseID** (Path, required): The ID of the course.
- **courseWorkID** (Path, required): The ID of the coursework item.
- **key** (Query, optional): The API key for accessing the service.
- **fields** (Query, optional): A comma-separated list of fields to return in the response.
- **quotaUser** (Query, optional): An optional request- level quota override.

Example Code

```python
import requests

def get_coursework_attachments(course_id, course_work_id, access_token):
    """
    Fetches attachments for a specific course work item in a Google Classroom course.

    Parameters:
    course_id (str): The ID of the course.
    course_work_id (str): The ID of the course work item.
    access_token (str): The OAuth 2.0 access token.

    Returns:
    dict: The response JSON from the API.
```

```
    """
    url = f'https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/attachments'

    headers = {
        'Authorization': f'Bearer {access_token}'
    }

    try:
        response = requests.get(url, headers=headers)

        # Check if the response status code indicates success
        if response.status_code == 200:
            return response.json()
        else:
            # Log or handle the error appropriately
            print(f'Error: {response.status_code}')
            print(response.json())
            return None
    except requests.RequestException as e:
        # Handle network-related errors
        print(f'Request failed: {e}')
        return None

# Example usage:
course_id = '1234567890abcdef'
course_work_id = '0987654321fedcba'
access_token = 'ya29.Your_OAuth_2.0_Access_Token'

attachments = get_coursework_attachments(course_id, course_work_id, access_token)
if attachments:
    print('Attachments:', attachments)
else:
    print('Failed to retrieve attachments.')
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments/{attachmentId}

Description: Updates an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Common Use Cases
- Update the attachments associated with a specific coursework assignment.
- Modify metadata of existing attachments, such as title, description, or file type.
- Add new attachments to an existing coursework assignment.

Example Request
```http
PATCH /v1/classroom/courses/{course_id}/courseWork/{courseWork_id}/addOnAttachments.patch
Content-Type: application/json

{
  "attachments": [
    {
      "attachmentId": "12345",
      "title": "Updated Assignment Title",
      "description": "Updated description of the attachment"
    },
    {
      "newAttachment": {
        "title": "New Attachment",
        "url": "https://example.com/file",
        "mimeType": "application/pdf"
      }
    }
  ]
}
```

Common Parameters
- **course_id**: The unique identifier for the course.
- **courseWork_id**: The unique identifier for the coursework assignment.
- **attachments**: An array containing details of attachments to update or add.
  - **attachmentId**: The unique identifier for an existing attachment to update.
  - **title**: The new title for the attachment.
  - **description**: The new description for the attachment.
  - **newAttachment** (optional): An object to add a new attachment.
    - **title**: The title of the new attachment.

- **url**: The URL of the new attachment.
- **mimeType**: The MIME type of the new attachment.

Example Code

```python
import requests

# Define the API endpoint
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/attachments/{attachmentId}"

# Replace with your actual parameter values
course_id = "1234567890"
course_work_id = "1234567890"
attachment_id = "1234567890"
new_title = "Updated Attachment Title"
new_description = "Updated Attachment Description"

# Define the headers (include any necessary authentication)
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the payload with the new attachment details
payload = {
    "title": new_title,
    "description": new_description
}

# Make the PATCH request to update the attachment
try:
    response = requests.patch(
        url.format(courseId=course_id, courseWorkId=course_work_id, attachmentId=attachment_id),
        headers=headers,
        json=payload
    )

    # Check for successful response
    response.raise_for_status()

    # Print the response (if needed)
    print("Attachment updated successfully:", response.json())

except requests.exceptions.HTTPError as err:
    print(f"HTTP error occurred: {err}")
except Exception as err:
    print(f"An error occurred: {err}")
```

```
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.studentSubmissions.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments/{attachmentId}/studentSubmissions/{submissionId}

Description: Returns a student submission for an add-on attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description

Endpoint: GET classroom.courses.courseWork.addOnAttachments.studentSubmissions.get

Common Use Cases:
- Retrieve specific attachments associated with student submissions for a given piece of coursework.
- Verify the status and details of student submissions, including any attachments.
- Audit submission details for assessment or compliance purposes.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentSubmissionId}/addOnAttachments
Authorization: Bearer {access_token}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `courseWorkId` (string): The unique identifier for the coursework.
- `studentSubmissionId` (string): The unique identifier for the student submission.

Optional Parameters:
- `alt` (string): Data format for the response. Can be "json" or "media".
- `fields` (string): Selector specifying which fields to include in a partial response.

Example Code

```python
import requests

def get_student_submissions(course_id, course_work_id, access_token):
    # Define the API endpoint URL
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions"

    # Set up headers with the access token for authentication
```

```
    headers = {
        'Authorization': f'Bearer {access_token}'
    }

    try:
        # Make the GET request to the API
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Return the JSON response data
            return response.json()
        else:
            # Handle non-success responses
            response.raise_for_status()

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except Exception as err:
        print(f'Other error occurred: {err}')

    return None

# Example usage
course_id = '1234567890'
course_work_id = 'abcdefghij'
access_token = 'your_access_token_here'

submissions = get_student_submissions(course_id, course_work_id, access_token)
if submissions:
    print("Student submissions:", submissions)
else:
    print("Failed to retrieve student submissions.")
```

Google API Documentation

Endpoint: courses.courseWork.addOnAttachments.studentSubmissions.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{itemId}/addOnAttachments/{attachmentId}/studentSubmissions/{submissionId}

Description: Updates data associated with an add-on attachment submission. Requires the add-on to have been the original creator of the attachment and the attachment to have a positive `max_points` value set. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Endpoint: PATCH classroom.courses.courseWork.addOnAttachments.studentSubmissions.patch

Common Use Cases
- Updating the status of a student submission attachment.
- Modifying metadata associated with a student submission attachment.
- Adjusting the review comments for a specific submission attachment.

Example Request
```http
PATCH
https://classroom.googleapis.com/v1/courses/courseId/courseWork/courseWorkId/addOnAttachments/attachmentId/studentSubmissions/submissionId
Content-Type: application/json

{
  "status": "REVIEWED",
  "reviewComments": "Great work! Keep it up."
}
```

Common Parameters
- **courseId** (string, required): The unique identifier for the course.
- **courseWorkId** (string, required): The unique identifier for the coursework item.
- **attachmentId** (string, required): The unique identifier for the attachment.
- **submissionId** (string, required): The unique identifier for the student submission.
- **status** (string, optional): The new status of the attachment, such as "NEW", "REVIEWED", etc.
- **reviewComments** (string, optional): Comments or feedback related to the submission.

Example Code

```python
import requests

# Define the API endpoint and necessary parameters
url =
"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{courseWork_id}/studentSubmissions/{student
```

```
_id}/attachments/{attachment_id}"

# Replace these placeholders with actual values
course_id = "your_course_id"
courseWork_id = "your_course_work_id"
student_id = "your_student_id"
attachment_id = "your_attachment_id"

# Define the headers with authentication token
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the payload for the patch request
payload = {
    "title": "Updated Assignment Title",  # Example parameter
    "driveFile": {
        "title": "Updated Drive File Title",  # Example parameter
              "alternateLink":  "https://drive.google.com/file/d/{drive_file_id}/view?usp=sharing"    #  Example
parameter
    }
}

# Make the patch request
response = requests.patch(
    url.format(
        course_id=course_id,
        courseWork_id=courseWork_id,
        student_id=student_id,
        attachment_id=attachment_id
    ),
    headers=headers,
    json=payload
)

# Check for errors and handle them
if response.status_code == 200:
    print("Attachment updated successfully.")
elif response.status_code == 404:
    print("Resource not found.")
elif response.status_code == 400:
    print("Bad request. Please check the payload and parameters.")
else:
    print(f"Error: {response.status_code}")
    print(f"Response: {response.text}")
```

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.return

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}:return

Description: Returns a student submission. Returning a student submission transfers ownership of attached Drive files to the student and may also update the submission state. Unlike the Classroom application, returning a student submission does not set assignedGrade to the draftGrade value. Only a teacher of the course that contains the requested student submission may call this method. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, return the requested student submission, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

Endpoint: POST classroom.courses.courseWork.studentSubmissions.return

Common Use Cases:
- Retrieving a list of student submissions for a specific piece of coursework.
- Checking the status of student submissions for evaluation purposes.
- Accessing submission details to provide feedback or grades to students.

Example Request:
```json
POST
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentId}/returns
Content-Type: application/json

{
  "returnStatus": "RETURNED"
}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `courseWorkId` (string): The unique identifier for the coursework.
- `studentId` (string): The unique identifier for the student.
- `returnStatus` (string): The status of the submission return, typically "RETURNED" or "NOT_RETURNED".

Example Code

```python
import requests

# Define the API endpoint
```

```
url = "https://classroom.googleapis.com/v1/courses/COURSE_ID/courseWork/COURSEWORK_ID/studentSubmissions"

# Define the API key
api_key = 'YOUR_API_KEY'

# Define the headers with the API key
headers = {
    'Authorization': f'Bearer {api_key}'
}

# Define the parameters for the request
params = {
    'studentId': 'STUDENT_ID',
    'courseId': 'COURSE_ID'
}

# Send the GET request to the API endpoint
response = requests.get(url, headers=headers, params=params)

# Check if the request was successful
if response.status_code == 200:
    # Parse the JSON response
    submissions = response.json()
    # Print the submissions
    print(submissions)
else:
    # Handle errors
    print(f"Error: Unable to fetch submissions. Status code: {response.status_code}")
    print(f"Response: {response.text}")
```

Explanation:
- **URL**: The endpoint URL for the API, with placeholders for `COURSE_ID` and `COURSEWORK_ID`.
- **API Key**: The authentication key for the API.
- **Headers**: Authorization header with the API key.
- **Parameters**: Parameters for the request, including `studentId` and `courseId`.
- **Request**: Sends a GET request to the API endpoint with the specified headers and parameters.
- **Error Handling**: Checks the response status code to determine if the request was successful. If not, it prints an error message with the status code and response text.

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}

Description: Updates one or more fields of a student submission. See google.classroom.v1.StudentSubmission for details of which fields may be updated and who may change them. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding course work, if the user is not permitted to make the requested modification to the student submission, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

Technical Description for PATCH classroom.courses.courseWork.studentSubmissions.patch

Common Use Cases
- Updating the submission status of a student's coursework.
- Modifying details of a submitted assignment, such as resubmission or feedback.
- Correcting metadata associated with a student's submission.

Example Request
```json
PATCH https://api.example.com/classroom.courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions
Content-Type: application/json

{
  "status": "RETURNED"
}
```

Common Parameters
- **courseId**: The unique identifier for the course.
- **courseWorkId**: The unique identifier for the specific coursework or assignment.
- **status**: The new status for the student submission (e.g., "TURNED_IN", "RETURNED").
- **feedback**: Optional parameter to provide feedback on the submission.
- **resubmissionStatus**: Optional parameter to indicate if the submission is a resubmission (e.g., "PENDING", "ACCEPTED").
- **dueDate**: Optional parameter to update the due date for the submission.

Example Code

```python
import requests

# Define the API endpoint and access token
url =
```

```
"https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentId
}"
access_token = "your_access_token_here"

# Define the headers with authorization
headers = {
    "Authorization": f"Bearer {access_token}",
    "Content-Type": "application/json"
}

# Define the parameters for the request
courseId = "1234567890"
courseWorkId = "9876543210"
studentId = "student_email@example.com"
body = {
    "status": "RECLAIMED"
}

# Make the PATCH request
try:
    response = requests.patch(
        url.format(courseId=courseId, courseWorkId=courseWorkId, studentId=studentId),
        headers=headers,
        json=body
    )
    response.raise_for_status()  # Raise an error for bad status codes
    print("Request was successful!")
    print("Response:", response.json())
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except Exception as err:
    print(f"Other error occurred: {err}")
```

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions

Description: Returns a list of student submissions that the requester is permitted to view, factoring in the OAuth scopes of the request. `-` may be specified as the `course_work_id` to include student submissions for multiple course work items. Course students may only view their own work. Course teachers and domain administrators may view all student submissions. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.courseWork.studentSubmissions.list

Common Use Cases:
- Retrieving a list of all student submissions for a specific assignment.
- Checking the status of student submissions to identify any overdue or missing work.
- Reviewing student submissions to facilitate grading and feedback.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions?key={YOUR_API_KEY}
```

Common Parameters:
- **courseId** (Required): The identifier of the course.
- **courseWorkId** (Required): The identifier of the course work (assignment).
- **key** (Optional): Your API key for authentication.
- **pageSize** (Optional): Maximum number of student submissions to return per page.
- **pageToken** (Optional): Token to specify the next page of results in a multi-page response.
- **states** (Optional): Comma-separated list of states to filter submissions by (e.g., "TURNED_IN,RETURNED,NEW").
- **userId** (Optional): The identifier of the user whose submissions are to be retrieved.

Example Code

```python
import google.auth
from googleapiclient.discovery import build

def list_student_submissions(course_id, assignment_id):
    # Authenticate and build the service
    credentials, project = google.auth.default()
    service = build('classroom', 'v1', credentials=credentials)
```

```
    try:
        # Define the parameters
        params = {
            'courseId': course_id,  # Replace with actual course ID
            'assignmentId': assignment_id  # Replace with actual assignment ID
        }

        # Call the API endpoint and get the response
        response = service.courses().courseWork().studentSubmissions().list(**params).execute()

        # Process the response (example: print submissions)
        for submission in response.get('studentSubmissions', []):
            print(f"Student ID: {submission['userId']}, Submission ID: {submission['id']}")

    except Exception as e:
        # Handle errors
        print(f"An error occurred: {e}")

# Example usage
course_id = '1234567890'  # Replace with a real course ID
assignment_id = '0987654321'  # Replace with a real assignment ID
list_student_submissions(course_id, assignment_id)
```

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.reclaim

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}:reclaim

Description: Reclaims a student submission on behalf of the student that owns it. Reclaiming a student submission transfers ownership of attached Drive files to the student and updates the submission state. Only the student that owns the requested student submission may call this method, and only for a student submission that has been turned in. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, unsubmit the requested student submission, or for access errors. * `FAILED_PRECONDITION` if the student submission has not been turned in. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

POST classroom.courses.courseWork.studentSubmissions.reclaim

Common Use Cases:
- Reclaim a student submission for a coursework assignment.
- Restore ownership of a student submission to a teacher.
- Retrieve a submitted assignment to review or correct.

Example Request:
```http
POST
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentSubmissionId}/reclaim
Authorization: Bearer {access_token}
Content-Type: application/json

{
  "reclaimReason": "Teacher review required"
}
```

Common Parameters:
- **courseId (path):** The ID of the course.
- **courseWorkId (path):** The ID of the coursework.
- **studentSubmissionId (path):** The ID of the student submission.
- **reclaimReason (body):** A reason for reclaiming the submission (optional).

Example Code

```python
import requests
```

```
def reclaim_submission(course_id, assignment_id, submission_id, access_token):
    """
    Reclaims a student submission for an assignment.

    :param course_id: The ID of the course.
    :param assignment_id: The ID of the assignment.
    :param submission_id: The ID of the submission to reclaim.
    :param access_token: The OAuth 2.0 access token.
    :return: The response from the API.
    """
                                                                                    url                    =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{assignment_id}/studentSubmissions/{submis
sion_id}:reclaim"

    headers = {
        'Authorization': f'Bearer {access_token}'
    }

    try:
        response = requests.post(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()  # Return the JSON response
    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except Exception as err:
        print(f'Other error occurred: {err}')

# Example usage:
course_id = '1234567890'
assignment_id = '0987654321'
submission_id = '1122334455'
access_token = 'your_oauth_access_token_here'

reclaim_response = reclaim_submission(course_id, assignment_id, submission_id, access_token)
if reclaim_response:
    print("Reclaim submission response:", reclaim_response)
```

This example demonstrates how to reclaim a student submission using the `classroom.courses.courseWork.studentSubmissions.reclaim` endpoint. It includes error handling for HTTP errors and other exceptions, and returns the JSON response from the API if the request is successful.

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.turnIn

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}:turnIn

Description: Turns in a student submission. Turning in a student submission transfers ownership of attached Drive files to the teacher and may also update the submission state. This may only be called by the student that owns the specified student submission. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, turn in the requested student submission, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

Endpoint: POST classroom.courses.courseWork.studentSubmissions.turnIn

Common Use Cases:
- Students submitting assignments.
- Teachers submitting work on behalf of students.
- Automated systems submitting assignments via APIs.

Example Request:
```json
POST
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentId}/turnIn
Content-Type: application/json

{
  "assignmentSubmission": {
    "driveFile": {
      "driveFileId": "1234567890abcdefghijklmnopqrstuvwxyz"
    },
    "linkedCourseWorkId": "courseWorkId",
    "alternateLink": "https://example.com/submission"
  }
}
```

Common Parameters:
- **courseId**: The ID of the course.
- **courseWorkId**: The ID of the coursework item.
- **studentId**: The ID of the student.
- **assignmentSubmission**: A JSON object containing details of the submission.
  - **driveFile**: Information about the Google Drive file being submitted.
    - **driveFileId**: The ID of the Google Drive file.

- **linkedCourseWorkId**: The ID of the coursework the submission is linked to.
- **alternateLink**: An optional link to an external submission.

Example Code

```python
import requests

def turn_in_course_work(course_id, course_work_id, access_token, submission_text=None):
    """
    Turn in a student submission for a course work.

    :param course_id: ID of the course.
    :param course_work_id: ID of the course work.
    :param access_token: OAuth 2.0 access token.
    :param submission_text: Text of the submission.
    """
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/turnIn
"

    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }

    payload = {}
    if submission_text:
        payload['submission'] = {'content': submission_text}

    try:
        response = requests.post(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
access_token = 'your_access_token_here'
submission_text = 'Here is the submission text for the course work.'

turn_in_course_work(course_id, course_work_id, access_token, submission_text)
```

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.modifyAttachments

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}:modifyAttachments

Description: Modifies attachments of student submission. Attachments may only be added to student submissions belonging to course work objects with a `workType` of `ASSIGNMENT`. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work, if the user is not permitted to modify attachments on the requested student submission, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

Technical Description

Endpoint: POST classroom.courses.courseWork.studentSubmissions.modifyAttachments

Common Use Cases:
- Update the attachments for a student submission.
- Add new files to an existing submission.
- Remove specific attachments from a submission.

Example Request:
```http
POST
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{studentId}/modifyAttachments
Content-Type: application/json
Authorization: Bearer ya29...

{
  "addAttachments": [
    {
      "fileUrl": "https://drive.google.com/file/d/{fileId}/view?usp=sharing",
      "title": "New Assignment Attachment",
      "mimeType": "application/pdf"
    }
  ],
  "removeAttachmentIds": ["attachmentId1", "attachmentId2"]
}
```

Common Parameters:
- **courseId** (string): The ID of the course.
- **courseWorkId** (string): The ID of the course work.
- **studentId** (string): The ID of the student whose submission is being modified.

- **addAttachments** (array of objects): List of attachments to add, each containing `fileUrl`, `title`, and `mimeType`.
- **removeAttachmentIds** (array of strings): List of IDs of attachments to remove.

Example Code

```python
import requests

def modify_coursework_submission_attachments(course_id, course_work_id, student_id, attachment_id):
    # Define the API endpoint URL
                                                                                    url                     =
f'https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/{stude
nt_id}/modifyAttachments'

    # Define the headers with authorization
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
        'Content-Type': 'application/json'
    }

    # Define the payload with realistic parameter values
    payload = {
        "add": [
            {
                "attachmentId": attachment_id
            }
        ]
    }

    try:
        # Send the POST request to modify attachments
        response = requests.post(url, headers=headers, json=payload)

        # Check for a successful response
        if response.status_code == 200:
            print('Attachments modified successfully')
            return response.json()
        else:
            # Handle unsuccessful response
            print(f'Failed to modify attachments: {response.status_code}')
            print(response.json())
            return None
    except requests.exceptions.RequestException as e:
        # Handle any request exceptions
        print(f'An error occurred: {e}')
        return None

# Example usage
course_id = '1234567890'
```

```
course_work_id = '9876543210'
student_id = 'student_email@example.com'
attachment_id = 'attachment_12345'

modify_coursework_submission_attachments(course_id, course_work_id, student_id, attachment_id)
```

Google API Documentation

Endpoint: courses.courseWork.studentSubmissions.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{id}

Description: Returns a student submission. * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course, course work, or student submission or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or student submission does not exist.

AI-Generated Documentation

Technical Description

Endpoint: GET classroom.courses.courseWork.studentSubmissions.get

Common Use Cases:
- Retrieve a list of student submissions for a specific piece of coursework.
- Monitor student participation and performance.
- Provide feedback to students on their submissions.

Example Request:
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions?studentId={studentId}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `courseWorkId` (string): The unique identifier for the coursework.
- `studentId` (string): The unique identifier for the student whose submissions are being retrieved.
- `updateTimeMin` (string, optional): The minimum update time of the submissions to retrieve.
- `updateTimeMax` (string, optional): The maximum update time of the submissions to retrieve.

Example Code

```python
import google_auth
from googleapiclient.discovery import build

def get_student_submissions(course_id, assignment_id, student_id):
    """
    Fetches student submissions for a specific course and assignment.

    Args:
        course_id (str): The ID of the course.
        assignment_id (str): The ID of the assignment.
        student_id (str): The ID of the student.
```

```
    Returns:
        dict: The student submission details.
    """
    # Initialize credentials and build the service
    credentials = google_auth.get_credentials()
    service = build('classroom', 'v1', credentials=credentials)

    try:
        # Call the API endpoint
        submission = service.courses().courseWork().studentSubmissions().get(
            courseId=course_id,
            courseWorkId=assignment_id,
            studentId=student_id
        ).execute()

        # Return the submission details
        return submission
    except google_auth.exceptions.GoogleAuthError as auth_err:
        # Handle authentication errors
        print(f"Authentication Error: {auth_err}")
    except googleapiclient.errors.HttpError as http_err:
        # Handle HTTP errors
        print(f"HTTP Error: {http_err}")
    except Exception as err:
        # Handle any other errors
        print(f"An error occurred: {err}")

    return None

# Example usage
course_id = '1234567890'  # Replace with a real course ID
assignment_id = '0987654321'  # Replace with a real assignment ID
student_id = '1122334455'  # Replace with a real student ID

submission = get_student_submissions(course_id, assignment_id, student_id)
if submission:
    print("Submission details:", submission)
else:
    print("Failed to retrieve submission details.")
```

Google API Documentation

Endpoint: courses.courseWork.rubrics.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics/{id}

Description: Deletes a rubric. The requesting user and course owner must have rubrics creation capabilities. For details, see [licensing requirements](https://developers.google.com/classroom/rubrics/limitations#license-requirements). This request must be made by the Google Cloud console of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding rubric. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project didn't create the corresponding rubric, or if the requesting user isn't permitted to delete the requested rubric. * `NOT_FOUND` if no rubric exists with the requested ID or the user does not have access to the course, course work, or rubric. * `INVALID_ARGUMENT` if grading has already started on the rubric.

AI-Generated Documentation

Technical Description for DELETE classroom.courses.courseWork.rubrics.delete

Common Use Cases:
- Delete a specific rubric associated with a coursework assignment.
- Remove outdated or incorrect rubric criteria for an assignment.
- Simplify the grading process by removing unnecessary rubrics.

Example Request:
```http
DELETE https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics/{rubricId}
Authorization: Bearer {YOUR_ACCESS_TOKEN}
```

Common Parameters:
- `courseId` (String): The unique identifier for the course.
- `courseWorkId` (String): The unique identifier for the coursework assignment.
- `rubricId` (String): The unique identifier for the rubric to be deleted.

Example Code

```python
import google_auth_oauthlib.flow
import googleapiclient.discovery

# Define the scopes required for the API
SCOPES = ['https://www.googleapis.com/auth/classroom.coursework.rubrics']

# Create a flow object for user authentication
flow = google_auth_oauthlib.flow.InstalledAppFlow.from_client_secrets_file(
    'client_secret.json', SCOPES)

# Run the flow to get credentials
```

```
credentials = flow.run_local_server(port=0)

# Build the service object
service = googleapiclient.discovery.build('classroom', 'v1', credentials=credentials)

def delete_coursework_rubric(course_id, coursework_id, rubric_id):
    """
    Deletes a rubric from a coursework in Google Classroom.

    :param course_id: The ID of the course.
    :param coursework_id: The ID of the coursework.
    :param rubric_id: The ID of the rubric to delete.
    """

    try:
        # Call the API to delete the rubric
        response = service.courses().courseWork().rubrics().delete(
            courseId=course_id,
            courseWorkId=coursework_id,
            id=rubric_id
        ).execute()

        # Print the response (if any)
        print(f"Rubric {rubric_id} deleted successfully: {response}")

    except googleapiclient.errors.HttpError as error:
        # Handle HTTP errors
        print(f"An error occurred: {error}")
    except Exception as e:
        # Handle any other exceptions
        print(f"Unexpected error: {e}")

# Example usage
course_id = '1234567890'
coursework_id = 'abcdefghij'
rubric_id = '1234567890abcdefgh'

delete_coursework_rubric(course_id, coursework_id, rubric_id)
```

Google API Documentation

Endpoint: courses.courseWork.rubrics.list

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics

Description: Returns a list of rubrics that the requester is permitted to view. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or course work doesn't exist or if the user doesn't have access to the corresponding course work.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.courseWork.rubrics.list

Common Use Cases:
- Retrieve a list of rubrics associated with a specific course work.
- Assess the criteria and scoring details for student assignments.
- Integrate rubric data into grading and feedback systems.

Example Request:
```
GET https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId**: The identifier for the course. (required)
- **courseWorkId**: The identifier for the course work. (required)
- **pageToken**: A token to retrieve the next set of results. (optional)
- **pageSize**: The maximum number of results to return. (optional)
- **fields**: Select specific fields to return. (optional)

Example Code

```python
from googleapiclient import discovery
from google.oauth2 import service_account

# Define the scope and credentials for the API
SCOPES = ['https://www.googleapis.com/auth/classroom.courses']
SERVICE_ACCOUNT_FILE = 'path/to/service-account-file.json'

# Initialize the service account credentials
credentials = service_account.Credentials.from_service_account_file(
    SERVICE_ACCOUNT_FILE, scopes=SCOPES)

# Create the API client
service = discovery.build('classroom', 'v1', credentials=credentials)
```

```
def list_course_work_rubrics(course_id):
    try:
        # Define the parameters for the API request
        params = {
            'courseId': course_id  # Example course ID
        }

        # Make the API request to list course work rubrics
        results = service.courses().courseWork().rubrics().list(**params).execute()

        # Process the results
        if 'rubrics' in results:
            for rubric in results['rubrics']:
                print(f"Rubric ID: {rubric['id']}, Title: {rubric['title']}")
        else:
            print("No rubrics found for the specified course.")

    except Exception as e:
        # Handle errors gracefully
        print(f"An error occurred: {e}")

# Example usage
course_id = 'YOUR_COURSE_ID'  # Replace with a real course ID
list_course_work_rubrics(course_id)
```

Google API Documentation

Endpoint: courses.courseWork.rubrics.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics/{id}

Description: Updates a rubric. See google.classroom.v1.Rubric for details of which fields can be updated. Rubric update capabilities are [limited](/classroom/rubrics/limitations) once grading has started. The requesting user and course owner must have rubrics creation capabilities. For details, see [licensing requirements](https://developers.google.com/classroom/rubrics/limitations#license-requirements). This request must be made by the Google Cloud console of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the parent course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project didn't create the corresponding course work, if the user isn't permitted to make the requested modification to the rubric, or for access errors. This error code is also returned if grading has already started on the rubric. * `INVALID_ARGUMENT` if the request is malformed and for the following request error: * `RubricCriteriaInvalidFormat` * `NOT_FOUND` if the requested course, course work, or rubric doesn't exist or if the user doesn't have access to the corresponding course work. * `INTERNAL` if grading has already started on the rubric.

AI-Generated Documentation

Endpoint: `PATCH classroom.courses.courseWork.rubrics.patch`

Common Use Cases:
- Update the criteria of a rubric.
- Modify the maximum points for a rubric.
- Change the description of a rubric.

Example Request:
```http
PATCH https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics/{rubricId}
Content-Type: application/json
Authorization: Bearer {access_token}

{
  "rubric": {
    "title": "Updated Rubric Title",
    "criteria": [
      {
        "id": "1",
        "description": "Updated criterion description",
        "points": 5
      }
    ]
  }
}
```

Common Parameters:
- **courseId** (string): The unique identifier for the course.

- **courseWorkId** (string): The unique identifier for the course work.
- **rubricId** (string): The unique identifier for the rubric.
- **rubric** (object): The updated rubric details.
  - **title** (string): The title of the rubric.
  - **criteria** (array): List of criteria for the rubric.
    - **id** (string): The unique identifier for the criterion.
    - **description** (string): The description of the criterion.
    - **points** (integer): The maximum points for the criterion.

Example Code

```python
import requests

# Define the API endpoint and parameters
api_url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics"
course_id = "1234567890"
course_work_id = "0987654321"
access_token = "YOUR_ACCESS_TOKEN"

# Define the headers and parameters for the request
headers = {
    "Authorization": f"Bearer {access_token}",
    "Content-Type": "application/json"
}

# Define the data to be sent in the PATCH request
data = {
    "rubric": {
        "title": "Updated Rubric Title",
        "description": "Updated rubric description for better understanding.",
        "criteria": [
            {
                "id": "1",
                "description": "Updated criterion description",
                "points": 10
            }
        ]
    }
}

try:
    # Make the PATCH request to update the rubric
    response = requests.patch(
        api_url.format(courseId=course_id, courseWorkId=course_work_id),
        headers=headers,
        json=data
    )
```

```
    # Check if the request was successful
    if response.status_code == 200:
        print("Rubric updated successfully.")
        print(response.json())
    else:
        print(f"Failed to update rubric. Status code: {response.status_code}")
        print(response.json())
except requests.exceptions.RequestException as e:
    # Handle any requests exceptions that occur
    print(f"An error occurred: {e}")
```

Google API Documentation

Endpoint: courses.courseWork.rubrics.get

HTTP Method: GET

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics/{id}

Description: Returns a rubric. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course, course work, or rubric doesn't exist or if the user doesn't have access to the corresponding course work.

AI-Generated Documentation

Technical Description

Common Use Cases
- Retrieve the grading rubric for a specific coursework assignment.
- Review the criteria used for grading an assignment.
- Understand the specific elements and point allocations for grading purposes.

Example Request
```
GET https://api.example.com/v1/classroom/courses/{courseId}/courseWork/{courseWorkId}/rubrics
```

Common Parameters
- **courseId**: The unique identifier for the course.
- **courseWorkId**: The unique identifier for the coursework assignment.
- **alt**: Specifies an alternative format for the response (e.g., JSON, atom, etc.).

Example Request with Placeholder Values
```
GET https://api.example.com/v1/classroom/courses/12345/courseWork/67890/rubrics?alt=json
```

Common Response
```json
{
  "kind": "classroom#rubric",
  "id": "rubric_id",
  "title": "Assignment Rubric",
  "description": "Grading rubric for the assignment",
  "criteria": [
    {
      "id": "criterion_id_1",
      "points": 10,
      "title": "Criterion Title 1",
      "description": "Criterion Description 1",
      "weight": 1
    }
```

```
    // Additional criteria...
  ]
}
```

Endpoint Description
The `GET classroom.courses.courseWork.rubrics.get` endpoint is used to retrieve the grading rubric for a specific coursework assignment within a specified course. By providing the course ID and coursework ID, you can access the rubric details, including the criteria, points allocated, and descriptions. This endpoint is useful for educational institutions to ensure transparency and consistency in grading assignments.

Example Code

```python
import requests

def get_course_work_rubric(course_id, course_work_id, key_file='path/to/your/keyfile.json'):
    """
    Fetches the rubric for a specific course work in a Google Classroom course.

    :param course_id: The ID of the course.
    :param course_work_id: The ID of the course work.
    :param key_file: Path to the Google API key file.
    :return: The rubric details if successful, otherwise an error message.
    """

    # Define the endpoint URL
    url = f'https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/rubrics'

    # Load the credentials from the key file
    try:
        with open(key_file, 'r') as f:
            key_data = json.load(f)
        token = key_data['access_token']  # Assuming key file contains 'access_token'
    except Exception as e:
        return f"Error loading key file: {e}"

    # Set up the headers with the access token
    headers = {
        'Authorization': f'Bearer {token}'
    }

    # Make the GET request to the API
    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        return f"HTTP error occurred: {http_err}"
```

```
    except requests.exceptions.ConnectionError as conn_err:
        return f"Connection error occurred: {conn_err}"
    except requests.exceptions.Timeout as timeout_err:
        return f"Timeout error occurred: {timeout_err}"
    except requests.exceptions.RequestException as req_err:
        return f"An error occurred: {req_err}"

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
rubric_details = get_course_work_rubric(course_id, course_work_id, 'path/to/your/keyfile.json')
print(rubric_details)
```

Google API Documentation

Endpoint: courses.courseWork.rubrics.create

HTTP Method: POST

Path: v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics

Description: Creates a rubric. The requesting user and course owner must have rubrics creation capabilities. For details, see [licensing requirements](https://developers.google.com/classroom/rubrics/limitations#license-requirements). For further details, see [Rubrics structure and known limitations](/classroom/rubrics/limitations). This request must be made by the Google Cloud console of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the parent course work item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user isn't permitted to create rubrics for course work in the requested course. * `INTERNAL` if the request has insufficient OAuth scopes. * `INVALID_ARGUMENT` if the request is malformed and for the following request error: * `RubricCriteriaInvalidFormat` * `NOT_FOUND` if the requested course or course work don't exist or the user doesn't have access to the course or course work. * `FAILED_PRECONDITION` for the following request error: * `AttachmentNotVisible`

AI-Generated Documentation

Technical Description for POST classroom.courses.courseWork.rubrics.create

Common Use Cases:
- Creating a rubric for a coursework assignment to standardize grading criteria.
- Defining assessment metrics for student projects or assignments.
- Ensuring consistency in evaluation across different assignments or courses.

Example Request:
```json
POST https://classroom.googleapis.com/v1/classroom.courses.courseWork.rubrics.create
Content-Type: application/json
Authorization: Bearer YOUR_ACCESS_TOKEN

{
  "courseWorkId": "1234567890",
  "title": "Sample Rubric",
  "description": "Rubric for evaluating student projects",
  "rubric": {
    "title": "Project Evaluation Rubric",
    "description": "Criteria for grading student projects",
    "scoringGuide": [
      {
        "title": "Originality",
        "description": "How original is the project?",
        "points": 10,
        "criteria": [
          {
            "title": "Highly Original",
            "description": "The project is highly original and innovative.",
            "points": 10
```

```
        },
        {
          "title": "Moderately Original",
          "description": "The project is moderately original.",
          "points": 7
        },
        {
          "title": "Somewhat Original",
          "description": "The project is somewhat original.",
          "points": 5
        }
      ]
    }
  ]
 }
}
```

Common Parameters:
- **courseWorkId**: The ID of the coursework for which the rubric is being created.
- **title**: The title of the rubric.
- **description**: A description of the rubric.
- **rubric**: An object containing the details of the rubric.
  - **title**: The title of the rubric.
  - **description**: A description of the rubric.
  - **scoringGuide**: An array of criteria and scoring details for the rubric.
    - **title**: The title of the criterion.
    - **description**: A description of the criterion.
    - **points**: The total points for this criterion.
    - **criteria**: An array of scoring levels within the criterion.
      - **title**: The title of the scoring level.
      - **description**: A description of the scoring level.
      - **points**: The points assigned to this scoring level.

Example Code

```python
import google.auth
import google.auth.transport.requests
import google.oauth2.credentials
from googleapiclient.discovery import build

# Authenticate and create the service
credentials, project = google.auth.default()
service = build('classroom', 'v1', credentials=credentials)

# Define the course ID and the rubric details
course_id = '1234567890abcdef'
rubric_title = 'Final Project Rubric'
```

```
rubric_description = 'Rubric for the final project evaluation.'
rubric_criteria = [
    {'title': 'Content', 'points': 50, 'description': 'Quality of content'},
    {'title': 'Creativity', 'points': 30, 'description': 'Level of creativity'},
    {'title': 'Presentation', 'points': 20, 'description': 'Presentation of the project'}
]

# Create the request body
request_body = {
    'title': rubric_title,
    'description': rubric_description,
    'criteria': rubric_criteria
}

try:
    # Call the API to create the rubric
    response = service.courses().courseWork().rubrics().create(
        courseId=course_id, body=request_body
    ).execute()

    # Print the response
    print('Rubric created successfully:', response)
except googleapiclient.errors.HttpError as error:
    # Handle errors
    print('An error occurred:', error)
except Exception as e:
    # Handle any other exceptions
    print('An unexpected error occurred:', e)
```

Google API Documentation

Endpoint: courses.students.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/students/{userId}

Description: Deletes a student of a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to delete students of this course or for access errors. * `NOT_FOUND` if no student of this course has the requested ID or if the course does not exist.

AI-Generated Documentation

Endpoint: DELETE classroom.courses.students.delete

Common Use Cases:
- Remove a student from a specific course.
- Manage student enrollment to ensure accurate class rosters.
- Handle administrative tasks related to student withdrawals.

Example Request:
```http
DELETE /classroom/courses/123/students/456
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId**: The unique identifier for the course from which the student will be removed.
- **studentId**: The unique identifier for the student to be removed from the course.
- **Authorization**: Bearer token for authentication.

Example Code

```python
import requests

# Define the API endpoint URL
url = "https://classroom.googleapis.com/v1/courses/{courseId}/students/{userId}"

# Define the headers with your API key
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your actual access token
}

# Define the parameters for the request
course_id = "example_course_id"  # Replace with actual course ID
user_id = "example_user_id"  # Replace with actual user ID

# Construct the full URL with the parameters
```

```
full_url = url.format(courseId=course_id, userId=user_id)

# Make the DELETE request to remove the student from the course
try:
    response = requests.delete(full_url, headers=headers)
    # Raise an exception if the request was unsuccessful
    response.raise_for_status()

    # Print the response from the server
    print("Student removed successfully:", response.json())

except requests.exceptions.HTTPError as http_err:
    # Handle HTTP errors
    print(f"HTTP error occurred: {http_err}")
except Exception as err:
    # Handle other errors
    print(f"An error occurred: {err}")
```

This script sends a DELETE request to the specified API endpoint to remove a student from a course. It includes proper error handling to manage potential HTTP errors and other exceptions. The URL is constructed dynamically using the provided course and user IDs. The response from the server is printed if the request is successful.

Google API Documentation

Endpoint: courses.students.list

HTTP Method: GET

Path: v1/courses/{courseId}/students

Description: Returns a list of students of this course that the requester is permitted to view. This method returns the following error codes: * `NOT_FOUND` if the course does not exist. * `PERMISSION_DENIED` for access errors.

AI-Generated Documentation

Endpoint: GET classroom.courses.students.list

Common Use Cases:
- Retrieve a list of students enrolled in a specific course.
- Generate reports on student enrollment for administrative purposes.
- Verify which students are in a particular course for attendance tracking.

Example Request:
```
GET https://api.example.com/classroom/courses/123/students
```

Common Parameters:

- `course_id` (required): The unique identifier for the course whose students are to be listed.
- `page` (optional): Specifies the page number of the results to return (default is 1).
- `page_size` (optional): Specifies the number of results per page (default is 10).
- `include_details` (optional): A boolean flag to include additional student details in the response (e.g., contact information, grades).

Example Code

```python
import requests

# Define the API endpoint and parameters
url = "https://classroom.googleapis.com/v1/courses/{course_id}/students"
course_id = "1234567890"
access_token = "your_access_token_here"

# Set up the headers with the authorization token
headers = {
    "Authorization": f"Bearer {access_token}"
}

# Define the parameters for the request
params = {
    "pageSize": 10,  # Number of results per page
```

```
    "pageToken": ""  # Token for the next page of results, if any
}

# Function to list students in a course
def list_students_in_course():
    try:
        # Make the GET request to the API
        response = requests.get(url.format(course_id=course_id), headers=headers, params=params)

        # Raise an exception for HTTP errors
        response.raise_for_status()

        # Parse the JSON response
        students = response.json()

        # Print the students
        print(students)

        # If there are more pages, handle pagination
        next_page_token = response.headers.get("X-Goog-Api-Client-Resource-Name-Key")
        if next_page_token:
            params["pageToken"] = next_page_token
            print("There are more pages. Continue pagination manually.")

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Request error occurred: {req_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Call the function to list students
list_students_in_course()
```

This code defines a function `list_students_in_course` that makes a GET request to the `classroom.courses.students.list` API endpoint to retrieve a list of students in a specified course. It includes proper error handling for HTTP errors, request exceptions, and general exceptions. The code also includes a mechanism to handle pagination if there are more pages of results.

Google API Documentation

Endpoint: courses.students.create

HTTP Method: POST

Path: v1/courses/{courseId}/students

Description: Adds a user as a student of a course. Domain administrators are permitted to [directly add](https://developers.google.com/classroom/guides/manage-users) users within their domain as students to courses within their domain. Students are permitted to add themselves to a course using an enrollment code. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to create students in this course or for access errors. * `NOT_FOUND` if the requested course ID does not exist. * `FAILED_PRECONDITION` if the requested user's account is disabled, for the following request errors: * CourseMemberLimitReached * CourseNotModifiable * UserGroupsMembershipLimitReached * InactiveCourseOwner * `ALREADY_EXISTS` if the user is already a student or teacher in the course.

AI-Generated Documentation

Common Use Cases:
- Enrolling a student in a specific course.
- Registering multiple students for a particular class.
- Handling student enrollment during the course registration period.

Example Request:
```json
POST /classroom/courses/students/create
Content-Type: application/json

{
  "course_id": "12345",
  "student_ids": ["1001", "1002", "1003"],
  "enrollment_date": "2023-10-01"
}
```

Common Parameters:
- **course_id**: The unique identifier for the course in which the students are being enrolled.
- **student_ids**: An array of unique identifiers for the students to be enrolled in the course.
- **enrollment_date**: The date on which the students are being enrolled (optional, can be a future date).

Example Code

```python
import requests

def create_student_in_course(course_id, student_data):
    """
    Create a student in a course using the classroom.courses.students.create endpoint.

    Parameters:
```

```
    - course_id (str): The ID of the course.
    - student_data (dict): A dictionary containing student information.

    Returns:
    - dict: The response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/students"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",  # Replace with your actual access token
        "Content-Type": "application/json"
    }

    try:
        response = requests.post(url, headers=headers, json=student_data)
        response.raise_for_status()  # Raise an exception for 4xx/5xx status codes

        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")  # Handle HTTP errors
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")  # Handle other request errors
    except Exception as e:
        print(f"An unexpected error occurred: {e}")  # Handle any other unexpected errors

# Example usage
course_id = "course12345"
student_data = {
    "studentId": "student54321",  # Unique ID of the student
    "email": "student@example.com",  # Student's email address
    "courseRole": "STUDENT"  # Role of the student in the course
}

response = create_student_in_course(course_id, student_data)
print(response)
```

Google API Documentation

Endpoint: courses.students.get

HTTP Method: GET

Path: v1/courses/{courseId}/students/{userId}

Description: Returns a student of a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to view students of this course or for access errors. * `NOT_FOUND` if no student of this course has the requested ID or if the course does not exist.

AI-Generated Documentation

Description

The `GET classroom.courses.students.get` API endpoint is used to retrieve a list of students enrolled in a specific course. This endpoint is commonly utilized by educational platforms for tasks such as generating student reports, managing course rosters, and providing access to student performance data.

Common Use Cases
1. **Student List Retrieval:** Administrators or instructors can retrieve a list of students for a specific course.
2. **Course Management:** Educators can review and manage student enrollments for planning and administrative purposes.
3. **Performance Tracking:** Teachers can access student data to track progress and performance over time.

Example Request
```
GET /classroom/courses/{courseId}/students
```

Common Parameters
- **courseId** (Path Parameter): The unique identifier for the course. Required.
- **limit** (Query Parameter): The maximum number of students to return. Optional. Default is 100.
- **offset** (Query Parameter): The number of students to skip before starting to collect the result set. Optional. Default is 0.
- **sort_by** (Query Parameter): Field to sort the results by (e.g., 'name', 'enrollment_date'). Optional. Default is 'name'.
- **order** (Query Parameter): The order of the results ('asc' for ascending, 'desc' for descending). Optional. Default is 'asc'.

Example Code

```python
import requests

def get_classroom_students(course_id, access_token):
    # Define the API endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/students"

    # Define the headers with the access token for authorization
    headers = {
```

```
        "Authorization": f"Bearer {access_token}"
    }

    try:
        # Make the GET request to the API
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Parse the JSON response
            students = response.json()
            return students
        else:
            # Handle non-successful responses
            response.raise_for_status()
    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        # Handle any other request-related errors
        print(f"Request error occurred: {req_err}")

    # Return None if there was an error
    return None

# Example usage
course_id = "1234567890"  # Replace with a valid course ID
access_token = "your_access_token_here"  # Replace with a valid access token

students = get_classroom_students(course_id, access_token)
if students:
    print("Students retrieved:", students)
else:
    print("Failed to retrieve students.")
```

Google API Documentation

Endpoint: courses.topics.get

HTTP Method: GET

Path: v1/courses/{courseId}/topics/{id}

Description: Returns a topic. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or topic, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or topic does not exist.

AI-Generated Documentation

Endpoint: GET classroom.courses.topics.get

Common Use Cases
- Retrieve a list of topics for a specific course.
- Fetch topic details for generating course summaries or study guides.
- Integrate with learning management systems to display course content dynamically.

Example Request
```
GET /classroom/courses/{course_id}/topics
```

Placeholder	Value
`{course_id}`	12345

Common Parameters
- **course_id** (required, string): The unique identifier for the course.
- **page** (optional, integer): The page number for pagination, default is 1.
- **page_size** (optional, integer): The number of topics per page, default is 10.
- **include_details** (optional, boolean): If set to `true`, includes detailed information about each topic, default is `false`.

Example Code

```python
import requests

def get_course_topics(course_id, api_key):
    url = "https://classroom.googleapis.com/v1/courses/{}/topics".format(course_id)
    headers = {
        'Authorization': 'Bearer ' + api_key
    }
    params = {
        'pageSize': 10,  # Number of topics to retrieve per page
        'pageToken': None  # Token for pagination, set to None for the first request
    }
```

```
    try:
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()  # Raises HTTPError for bad responses
        data = response.json()
        return data
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")  # Log the error
    except Exception as err:
        print(f"An error occurred: {err}")  # Log the error

# Example usage:
course_id = '1234567890'  # Replace with a valid course ID
api_key = 'your_api_key_here'  # Replace with a valid API key
course_topics = get_course_topics(course_id, api_key)
print(course_topics)
```

Google API Documentation

Endpoint: courses.topics.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/topics/{id}

Description: Deletes a topic. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not allowed to delete the requested topic or for access errors. * `FAILED_PRECONDITION` if the requested topic has already been deleted. * `NOT_FOUND` if no course or topic exists with the requested ID.

AI-Generated Documentation

Endpoint: DELETE classroom.courses.topics.delete

Description:
This API endpoint is used to delete a specific topic from a course within a classroom. It is commonly used to remove outdated or irrelevant topics, reorganize course content, or purge unnecessary information.

Common Use Cases:
1. **Curriculum Update**: Deleting obsolete topics to update the course content.
2. **Content Management**: Removing specific topics to streamline the learning path.
3. **Error Correction**: Deleting incorrectly added topics.

Example Request:
```http
DELETE /classroom/courses/123/topics/456
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId**: The unique identifier for the course from which the topic will be deleted.
- **topicId**: The unique identifier for the topic to be deleted.
- **Authorization**: Bearer token for authentication.

Example Code

```python
import requests

def delete_course_topic(course_id, topic_id, api_key):
    """
    Deletes a topic from a course.

    Args:
    course_id (str): The ID of the course.
    topic_id (str): The ID of the topic to delete.
    api_key (str): The API key for authentication.
```

```
    Returns:
    dict: The response from the API.
    """

    # Define the API endpoint
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/topics/{topic_id}"

    # Set up the headers for the request
    headers = {
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }

    try:
        # Make the DELETE request
        response = requests.delete(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the response in JSON format
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        # Handle other errors
        print(f"An error occurred: {err}")

    return None

# Example usage
course_id = "1234567890"
topic_id = "abcdefghij"
api_key = "your_api_key_here"

response = delete_course_topic(course_id, topic_id, api_key)
if response:
    print("Topic deleted successfully:", response)
else:
    print("Failed to delete the topic.")
```

Google API Documentation

Endpoint: courses.topics.list

HTTP Method: GET

Path: v1/courses/{courseId}/topics

Description: Returns the list of topics that the requester is permitted to view. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

API Endpoint: GET classroom.courses.topics.list

Common Use Cases:
- Fetching a list of topics for a specific course to display in a student dashboard.
- Retrieving topics to populate a teacher's lesson planning interface.
- Updating course materials based on the available topics.

Example Request:
```
GET /classroom/courses/{courseId}/topics
Host: api.example.com
Authorization: Bearer {access_token}
```

Common Parameters:
- `courseId` (Path Parameter): The unique identifier for the course.
- `access_token` (Header Parameter): The authentication token for access control.
- `include_details` (Query Parameter, optional): If set to `true`, includes detailed information about each topic.
- `page` (Query Parameter, optional): The page number for pagination.
- `page_size` (Query Parameter, optional): The number of topics per page.

Example Code

```python
import requests

def list_topics(course_id, access_token):
    # Define the API endpoint URL
    url = "https://classroom.googleapis.com/v1/courses/{}/topics".format(course_id)

    # Define the headers with the access token for authentication
    headers = {
        "Authorization": f"Bearer {access_token}"
    }

    try:
        # Make the API request
```

```
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        response.raise_for_status()

        # Parse the JSON response
        topics = response.json().get('topics', [])

        # Return the list of topics
        return topics

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        # Handle any other errors
        print(f"An error occurred: {err}")

# Example usage
if __name__ == "__main__":
    course_id = "1234567890"
    access_token = "YOUR_ACCESS_TOKEN"
    topics = list_topics(course_id, access_token)
    print(topics)
```

Google API Documentation

Endpoint: courses.topics.create

HTTP Method: POST

Path: v1/courses/{courseId}/topics

Description: Creates a topic. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course, create a topic in the requested course, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `ALREADY_EXISTS` if there exists a topic in the course with the same name. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

Endpoint: POST classroom.courses.topics.create

Common Use Cases:
- Creating a new topic within an existing course to organize course content.
- Adding a new section to a course for structured learning.

Example Request:
```json
{
  "course_id": "12345",
  "topic_title": "Introduction to Algorithms",
  "topic_description": "An overview of fundamental algorithms and data structures."
}
```

Common Parameters:
- `course_id` (string): The unique identifier for the course in which the topic will be created.
- `topic_title` (string): The title of the new topic.
- `topic_description` (string, optional): A brief description of the topic.

Example Code

```python
import requests

def create_topic(api_url, api_key, course_id, title, description):
    """
    Create a new topic in a course using the classroom.courses.topics.create API.

    :param api_url: The base URL for the API.
    :param api_key: The API key for authentication.
    :param course_id: The ID of the course.
    :param title: The title of the new topic.
    :param description: The description of the new topic.
    :return: The response from the API.
    """
```

```
    # Define the endpoint URL
    endpoint = f"{api_url}/v1/courses/{course_id}/topics"

    # Define the headers with the API key
    headers = {
        'Authorization': f'Bearer {api_key}',
        'Content-Type': 'application/json'
    }

    # Define the payload with the topic details
    payload = {
        'title': title,
        'description': description
    }

    try:
        # Make the API request
        response = requests.post(endpoint, headers=headers, json=payload)

        # Check if the request was successful
        if response.status_code == 201:
            return response.json()
        else:
            # Handle different HTTP status codes
            if response.status_code == 400:
                print("Bad Request: Check the payload and try again.")
            elif response.status_code == 401:
                print("Unauthorized: Invalid API key.")
            elif response.status_code == 404:
                print("Not Found: Course ID not found.")
            else:
                print(f"Unexpected error: {response.status_code}")
            return None

    except requests.exceptions.RequestException as e:
        # Handle network-related errors
        print(f"Request failed: {e}")
        return None

# Example usage
api_url = "https://api.example.com"
api_key = "your_api_key_here"
course_id = "12345"
title = "New Topic"
description = "This is a new topic for the course."

response = create_topic(api_url, api_key, course_id, title, description)
if response:
    print("Topic created successfully:", response)
```

```
else:
    print("Failed to create topic.")
```

Google API Documentation

Endpoint: courses.topics.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/topics/{id}

Description: Updates one or more fields of a topic. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding topic or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `FAILED_PRECONDITION` if there exists a topic in the course with the same name. * `NOT_FOUND` if the requested course or topic does not exist

AI-Generated Documentation

Technical Description for PATCH classroom.courses.topics.patch

Common Use Cases:
- Updating the title or description of a specific topic within a course.
- Modifying the sequence or hierarchy of topics.
- Adding or removing attachments related to a topic.

Example Request:
```http
PATCH /classroom/courses/123/topics/456
Content-Type: application/json

{
  "title": "New Topic Title",
  "description": "Updated description of the topic."
}
```

Common Parameters:
- **title** (string, optional): The new title for the topic.
- **description** (string, optional): The updated description for the topic.
- **sequence** (integer, optional): The new sequence number for the topic.
- **attachments** (array, optional): A list of attachments to add or remove from the topic.

Example Code

```python
import requests

def update_topic_in_course(course_id, topic_id, new_topic_details):
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/topics/{topic_id}"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your actual access token
    }
    payload = new_topic_details
```

```
    try:
        response = requests.patch(url, json=payload, headers=headers)
        response.raise_for_status()  # Raises an HTTPError for bad responses (4xx and 5xx)

        # Parse and return the updated topic details
        updated_topic = response.json()
        return updated_topic

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = "12345"  # Replace with your actual course ID
topic_id = "67890"    # Replace with your actual topic ID
new_topic_details = {
    "topic": {
        "title": "New Topic Title",
        "description": "This is the updated description for the topic."
    }
}

updated_topic = update_topic_in_course(course_id, topic_id, new_topic_details)
if updated_topic:
    print("Topic updated successfully:", updated_topic)
```

Note: Replace `"YOUR_ACCESS_TOKEN"`, `"12345"`, and `"67890"` with your actual access token, course ID, and topic ID respectively.

Google API Documentation

Endpoint: courses.teachers.list

HTTP Method: GET

Path: v1/courses/{courseId}/teachers

Description: Returns a list of teachers of this course that the requester is permitted to view. This method returns the following error codes: * `NOT_FOUND` if the course does not exist. * `PERMISSION_DENIED` for access errors.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.teachers.list

Common Use Cases:
- Fetching a list of teachers assigned to a specific course.
- Retrieving teacher information for administrative or reporting purposes.
- Identifying teachers for course-related communications or notifications.

Example Request:
```http
GET /classroom/courses/{courseId}/teachers?page=1&limit=20
```

Common Parameters:
- **courseId (path)**: The unique identifier for the course to list teachers from.
- **page (query)**: The page number for pagination (default is 1).
- **limit (query)**: The number of records to return per page (default is 20).
- **sort (query)**: The field to sort the list by (e.g., name, email).
- **order (query)**: The sort order (e.g., asc, desc).

Example Code

```python
import requests

def list_teachers(course_id):
    # API endpoint URL
    url = "https://classroom.googleapis.com/v1/courses/{}/teachers".format(course_id)

    # Headers for the request
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with a valid access token
    }

    try:
        # Make the GET request to the API
        response = requests.get(url, headers=headers)

        # Check if the request was successful
```

```
        response.raise_for_status()

        # Parse the JSON response
        teachers = response.json()

        # Return the list of teachers
        return teachers

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Error occurred: {req_err}")

# Example usage
course_id = "YOUR_COURSE_ID"  # Replace with a valid course ID
teachers = list_teachers(course_id)
if teachers:
    print(teachers)
```

Google API Documentation

Endpoint: courses.teachers.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/teachers/{userId}

Description: Removes the specified teacher from the specified course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to delete teachers of this course or for access errors. * `NOT_FOUND` if no teacher of this course has the requested ID or if the course does not exist. * `FAILED_PRECONDITION` if the requested ID belongs to the primary teacher of this course. * `FAILED_PRECONDITION` if the requested ID belongs to the owner of the course Drive folder. * `FAILED_PRECONDITION` if the course no longer has an active owner.

AI-Generated Documentation

Technical Description for DELETE classroom.courses.teachers.delete

Common Use Cases:
- Remove a teacher from a specific course.
- Update the course management structure by deleting a teacher's assignment.
- Ensure data integrity by removing incorrect or outdated teacher assignments.

Example Request:
```
DELETE /classroom/courses/teachers/123
```

Common Parameters:
- **course_id** (Path Parameter): The unique identifier for the course from which the teacher is to be deleted.
- **teacher_id** (Path Parameter): The unique identifier for the teacher to be removed from the course.

Example Request Body:
```
None
```

Note: This endpoint may require authorization headers, such as `Authorization: Bearer <token>`, depending on the implementation.

Example Code

```python
import requests

def delete_teacher_from_course(course_id, teacher_id, api_key):
    """
    Deletes a teacher from a course using the classroom.courses.teachers.delete API endpoint.

    :param course_id: The ID of the course from which the teacher will be deleted.
```

```
    :param teacher_id: The ID of the teacher to be deleted from the course.
    :param api_key: The API key for authentication.
    :return: Response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/teachers/{teacher_id}"
    headers = {
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }

    try:
        response = requests.delete(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Return the response JSON if the deletion was successful
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
        return None
    except Exception as err:
        print(f"An error occurred: {err}")
        return None

# Example usage
api_key = "your_api_key_here"
course_id = "1234567890"
teacher_id = "teacher_email@example.com"

response = delete_teacher_from_course(course_id, teacher_id, api_key)
if response:
    print("Teacher successfully deleted from the course.")
else:
    print("Failed to delete the teacher from the course.")
```

Google API Documentation

Endpoint: courses.teachers.create

HTTP Method: POST

Path: v1/courses/{courseId}/teachers

Description: Creates a teacher of a course. Domain administrators are permitted to [directly add](https://developers.google.com/classroom/guides/manage-users) users within their domain as teachers to courses within their domain. Non-admin users should send an Invitation instead. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to create teachers in this course or for access errors. * `NOT_FOUND` if the requested course ID does not exist. * `FAILED_PRECONDITION` if the requested user's account is disabled, for the following request errors: * CourseMemberLimitReached * CourseNotModifiable * CourseTeacherLimitReached * UserGroupsMembershipLimitReached * InactiveCourseOwner * `ALREADY_EXISTS` if the user is already a teacher or student in the course.

AI-Generated Documentation

Concise Technical Description

Endpoint: POST classroom.courses.teachers.create

Common Use Cases:
- Adding a new teacher to a specific course.
- Assigning a teacher to multiple courses.
- Updating the teacher's role or responsibilities within a course.

Example Request:
```json
POST /classroom.courses/teachers/create
Content-Type: application/json

{
  "course_id": "12345",
  "teacher_id": "67890",
  "role": "Instructor",
  "additional_info": {
    "contact_email": "teacher@example.com",
    "office_hours": "M-F 9 AM - 5 PM"
  }
}
```

Common Parameters:
- `course_id` (string): The unique identifier for the course.
- `teacher_id` (string): The unique identifier for the teacher.
- `role` (string): The role of the teacher within the course (e.g., Instructor, TA).
- `additional_info` (object): Optional additional information about the teacher (e.g., contact details, office hours).

Example Code

```python
import requests

def create_teacher_in_course(course_id, first_name, last_name, email):
    """
    Function to create a teacher in a specific course using the classroom.courses.teachers.create endpoint.

    :param course_id: The ID of the course.
    :param first_name: The first name of the teacher.
    :param last_name: The last name of the teacher.
    :param email: The email of the teacher.
    :return: Response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/teachers"
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
        'Content-Type': 'application/json'
    }
    payload = {
        "emailAddress": email,
        "first_name": first_name,
        "last_name": last_name
    }

    try:
        response = requests.post(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()  # Return the JSON response from the API
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"An error occurred: {req_err}")
    return None

# Example usage
course_id = "1234567890"
first_name = "John"
last_name = "Doe"
email = "john.doe@example.com"

result = create_teacher_in_course(course_id, first_name, last_name, email)
if result:
    print("Teacher created successfully:", result)
else:
    print("Failed to create teacher.")
```

```
```

Google API Documentation

Endpoint: courses.teachers.get

HTTP Method: GET

Path: v1/courses/{courseId}/teachers/{userId}

Description: Returns a teacher of a course. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to view teachers of this course or for access errors. * `NOT_FOUND` if no teacher of this course has the requested ID or if the course does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.teachers.get

Common Use Cases:
- Retrieve a list of teachers assigned to a specific course.
- Verify the teachers associated with a particular course for administrative purposes.
- Populate a dashboard or interface with teacher information for students or parents.

Example Request:
```http
GET /classroom.courses.teachers.get?course_id=12345
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **course_id** (required): The unique identifier for the course for which teacher details are requested.
- **access_token** (optional): The access token required for authentication. If not provided in the header, it should be included as a parameter.
- **include_details** (optional): A boolean flag to include additional teacher details (e.g., contact information, profiles) in the response. Default is `false`.

Example Response:
```json
{
  "status": "success",
  "course_id": "12345",
  "teachers": [
    {
      "teacher_id": "T1",
      "name": "John Doe",
      "email": "john.doe@example.com",
      "subjects": ["Math", "Science"]
    },
    {
      "teacher_id": "T2",
      "name": "Jane Smith",
      "email": "jane.smith@example.com",
```

```
    "subjects": ["History", "Geography"]
  }
 ]
}
```

Example Code

```python
import requests

def get_teachers(course_id, api_key):
    # Define the API endpoint and parameters
    url = "https://api.learningplatform.com/classroom/courses/teachers/get"
    params = {
        "course_id": course_id,
        "api_key": api_key
    }

    try:
        # Make the API request
        response = requests.get(url, params=params)

        # Raise an exception for HTTP errors
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Extract and return the list of teachers
        teachers = data.get("teachers", [])

        return teachers

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = "12345"
api_key = "your_api_key_here"
teachers = get_teachers(course_id, api_key)
if teachers:
    for teacher in teachers:
        print(f"Teacher Name: {teacher['name']}, Email: {teacher['email']}")
```

In this example, the `get_teachers` function takes `course_id` and `api_key` as parameters, makes a GET request

to the API endpoint, and handles possible errors. The function returns a list of teachers if the request is successful. The example usage demonstrates how to call the function and print the teacher details.

Google API Documentation

Endpoint: courses.announcements.get

HTTP Method: GET

Path: v1/courses/{courseId}/announcements/{id}

Description: Returns an announcement. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or announcement, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or announcement does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.announcements.get

Common Use Cases:
- Retrieve a list of announcements for a specific course.
- Display announcements on a course dashboard.
- Integrate course announcements into a student portal or learning management system.

Example Request:
```
GET https://api.example.com/v1/classroom.courses.announcements.get
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
CourseId: 12345
```

Common Parameters:
- **Authorization**: Bearer token for API authentication.
- **CourseId**: Unique identifier for the course.
- **Limit**: (Optional) The maximum number of announcements to return.
- **Offset**: (Optional) The starting point for the announcements list.
- **SortBy**: (Optional) The field to sort the announcements by (e.g., date, title).

Example Code

```python
import requests

def get_classroom_announcements(course_id, access_token):
    # Define the API endpoint
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements"

    # Define the headers with the access token
    headers = {
        'Authorization': f'Bearer {access_token}'
    }
```

```
    try:
        # Make the GET request to the API
        response = requests.get(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Parse the JSON response
        announcements = response.json()

        # Return the list of announcements
        return announcements['announcements']

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        print(f"An error occurred: {err}")
    return None

# Example usage
course_id = "example_course_id"
access_token = "your_access_token"
announcements = get_classroom_announcements(course_id, access_token)
if announcements:
    for announcement in announcements:
        print(announcement['text'])
```

This code defines a function `get_classroom_announcements` that takes `course_id` and `access_token` as parameters. It constructs the API endpoint URL, sets the necessary headers for authorization, and makes a GET request to retrieve the announcements for the specified course. Proper error handling is included to manage HTTP errors and other exceptions. The announcements are printed if the request is successful.

Google API Documentation

Endpoint: courses.announcements.create

HTTP Method: POST

Path: v1/courses/{courseId}/announcements

Description: Creates an announcement. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course, create announcements in the requested course, share a Drive attachment, or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist. * `FAILED_PRECONDITION` for the following request error: * AttachmentNotVisible

AI-Generated Documentation

Endpoint: POST classroom.courses.announcements.create

Common Use Cases
- Creating a new announcement for a specific course.
- Notifying students about upcoming deadlines, events, or general updates.
- Broadcasting important information to all students enrolled in a course.

Example Request
```json
{
  "course_id": "12345",
  "title": "Important Class Update",
  "content": "Please note that the next class will be held online.",
  "author": "Teacher Name"
}
```

Common Parameters
- **course_id** (string): The unique identifier for the course.
- **title** (string): The title of the announcement.
- **content** (string): The main body of the announcement.
- **author** (string): The name of the person creating the announcement.

Example Code

```python
import requests

def create_announcement(course_id, text, headers):
    """
    Create an announcement in a Google Classroom course.

    Parameters:
    - course_id (str): The ID of the course to create the announcement in.
    - text (str): The text of the announcement.
    - headers (dict): Authorization headers including the access token.
```

```
    Returns:
    - dict: The response from the API or an error message.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements"
    payload = {
        "text": text
    }

    try:
        response = requests.post(url, json=payload, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        return {"error": f"HTTP error occurred: {http_err}"}
    except Exception as err:
        return {"error": f"An error occurred: {err}"}

# Example usage:
# headers = {
#     "Authorization": "Bearer YOUR_ACCESS_TOKEN"
# }
# course_id = "1234567890"
# announcement_text = "Welcome to the new class. Please review the syllabus."
# result = create_announcement(course_id, announcement_text, headers)
# print(result)
```

Google API Documentation

Endpoint: courses.announcements.getAddOnContext

HTTP Method: GET

Path: v1/courses/{courseId}/announcements/{itemId}/addOnContext

Description: Gets metadata for Classroom add-ons in the context of a specific post. To maintain the integrity of its own data and permissions model, an add-on should call this to validate query parameters and the requesting user's role whenever the add-on is opened in an [iframe](https://developers.google.com/classroom/add-ons/get-started/iframes/iframes-overview). This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.announcements.getAddOnContext

Common Use Cases:
- Retrieve contextual information for an announcement in a specific course.
- Integrate third-party applications to enhance classroom interactions.
- Automate processes that require announcement context, such as notifications or reporting.

Example Request:
```
GET https://classroom.googleapis.com/v1/courses/{courseId}/announcements/{announcementId}/getAddOnContext
Authorization: Bearer {access_token}
```

Common Parameters:
- **courseId (Path Parameter)**: The unique identifier for the course.
- **announcementId (Path Parameter)**: The unique identifier for the announcement.
- **access_token (Header Parameter)**: The OAuth 2.0 access token for authorization.

Google API Documentation

Endpoint: courses.announcements.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/announcements/{id}

Description: Deletes an announcement. This request must be made by the Developer Console project of the [OAuth client ID](https://support.google.com/cloud/answer/6158849) used to create the corresponding announcement item. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding announcement, if the requesting user is not permitted to delete the requested course or for access errors. * `FAILED_PRECONDITION` if the requested announcement has already been deleted. * `NOT_FOUND` if no course exists with the requested ID.

AI-Generated Documentation

API Endpoint: DELETE classroom.courses.announcements.delete

Common Use Cases:
1. Remove an announcement from a specific course.
2. Delete outdated or inaccurate announcements.
3. Ensure that the announcement list remains relevant and up-to-date.

Example Request:
```
DELETE /classroom/courses/{courseId}/announcements/{announcementId}
Host: example.com
Authorization: Bearer {access_token}
```

Common Parameters:
- `courseId` (Path Parameter): The unique identifier for the course from which the announcement is to be deleted.
- `announcementId` (Path Parameter): The unique identifier for the announcement to be deleted.
- `Authorization` (Header): The authorization token to authenticate the request.

Example Code

```python
import requests

def delete_announcement(course_id, announcement_id, access_token):
    """
    Delete an announcement from a course.

    :param course_id: The ID of the course.
    :param announcement_id: The ID of the announcement to delete.
    :param access_token: The access token for authentication.
    :return: The response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}"
```

```
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }

    try:
        response = requests.delete(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        print("Announcement deleted successfully.")
        return response.json()  # Return the JSON response if needed
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Request error occurred: {req_err}")

    return None

# Example usage
course_id = 'course_12345'
announcement_id = 'announcement_67890'
access_token = 'your_access_token_here'

delete_announcement(course_id, announcement_id, access_token)
```

Google API Documentation

Endpoint: courses.announcements.list

HTTP Method: GET

Path: v1/courses/{courseId}/announcements

Description: Returns a list of announcements that the requester is permitted to view. Course students may only view `PUBLISHED` announcements. Course teachers and domain administrators may view all announcements. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course does not exist.

AI-Generated Documentation

Technical Description for GET classroom.courses.announcements.list

Common Use Cases:
- Retrieve a list of announcements for a specific course.
- Monitor updates and notifications for a course.
- Integrate with course management systems to display announcements.

Example Request:
```
GET /classroom.courses.announcements.list?course_id=12345
```

Common Parameters:
- **course_id (required)**: The unique identifier for the course.
- **page_token (optional)**: A token to fetch the next set of results in case of pagination.
- **max_results (optional)**: The maximum number of results to return. Default is 100.

Example Code

```python
import requests

# Define the API endpoint and parameters
api_endpoint = 'https://classroom.googleapis.com/v1/courses/{course_id}/announcements'
course_id = '1234567890'  # Replace with a valid course ID
page_size = 10  # Number of announcements to retrieve per page

# Define the headers for authentication
headers = {
    'Authorization': 'Bearer YOUR_ACCESS_TOKEN'  # Replace with your access token
}

# Define the parameters for the request
params = {
    'pageSize': page_size
```

```
}

# Make the API request
try:
    response = requests.get(api_endpoint.format(course_id=course_id), headers=headers, params=params)
    response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
    announcements = response.json()
    print(announcements)
except requests.exceptions.HTTPError as http_err:
    print(f'HTTP error occurred: {http_err}')  # Handle HTTP errors
except Exception as err:
    print(f'Other error occurred: {err}')  # Handle other potential errors
```

This script makes a GET request to the `classroom.courses.announcements.list` endpoint to list announcements for a specific course. It includes parameters for the course ID and the number of announcements to retrieve per page. Error handling is included to manage HTTP errors and other exceptions.

Google API Documentation

Endpoint: courses.announcements.modifyAssignees

HTTP Method: POST

Path: v1/courses/{courseId}/announcements/{id}:modifyAssignees

Description: Modifies assignee mode and options of an announcement. Only a teacher of the course that contains the announcement may call this method. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access the requested course or course work or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if the requested course or course work does not exist.

AI-Generated Documentation

Technical Description for `POST classroom.courses.announcements.modifyAssignees`

Common Use Cases
1. **Update Course Announcement Assignees**: Modify the list of students or groups assigned to an announcement.
2. **Reassign Announcements**: Change the assignees of an announcement after an administrative decision or course changes.
3. **Batch Assignments**: Update multiple announcements' assignees in a single API call.

Example Request
```json
{
  "announcementId": "12345",
  "newAssignees": ["student123", "student456"],
  "courseId": "course789"
}
```

Common Parameters
- **announcementId**: The unique identifier for the announcement being modified.
- **newAssignees**: A list of user IDs or group IDs representing the new assignees for the announcement.
- **courseId**: The unique identifier for the course associated with the announcement.

Example Code

```python
import requests

# Define the API endpoint and the necessary headers
url = "https://classroom.googleapis.com/v1/courses/1234567890/announcements/announcementId/modifyAssignees"

# Define the headers for the request
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}
```

```
# Define the payload with realistic parameter names and values
payload = {
    "addAssignees": [
        {"userId": "user1@example.com"},
        {"userId": "user2@example.com"}
    ],
    "removeAssignees": [
        {"userId": "user3@example.com"}
    ]
}

# Make the API request
response = requests.post(url, headers=headers, json=payload)

# Check for HTTP errors
if response.status_code == 200:
    print("Announcement assignees modified successfully.")
    print("Response:", response.json())
elif response.status_code == 400:
    print("Bad Request: Invalid input parameters.")
    print("Error:", response.json())
elif response.status_code == 401:
    print("Unauthorized: Invalid or missing access token.")
    print("Error:", response.json())
elif response.status_code == 403:
    print("Forbidden: Insufficient permissions.")
    print("Error:", response.json())
elif response.status_code == 404:
    print("Not Found: Course or announcement not found.")
    print("Error:", response.json())
elif response.status_code == 500:
    print("Internal Server Error: Something went wrong on the server.")
    print("Error:", response.json())
else:
    print("Unexpected error occurred.")
    print("Error:", response.json())
```

Google API Documentation

Endpoint: courses.announcements.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/announcements/{id}

Description: Updates one or more fields of an announcement. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting developer project did not create the corresponding announcement or for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `FAILED_PRECONDITION` if the requested announcement has already been deleted. * `NOT_FOUND` if the requested course or announcement does not exist

AI-Generated Documentation

Common Use Cases:
- Update the content of an existing announcement.
- Modify the due date or visibility settings of an announcement.
- Correct any errors in an announcement's details.

Example Request:
```json
PATCH /classroom.courses.announcements.patch
Content-Type: application/json

{
  "announcement_id": "12345",
  "title": "Updated Homework Assignment",
  "content": "Please complete the updated homework by Friday.",
  "due_date": "2023-12-15",
  "visibility": "public"
}
```

Common Parameters:
- **announcement_id** (string): The unique identifier for the announcement to be updated.
- **title** (string, optional): The new title for the announcement.
- **content** (string, optional): The updated content or description of the announcement.
- **due_date** (string, optional): The new due date for the announcement, formatted as YYYY-MM-DD.
- **visibility** (string, optional): The new visibility setting for the announcement (e.g., "public", "private").

Example Code

```python
import requests

# Define the base URL for the API
base_url = "https://classroom.googleapis.com/v1"

# Define the course ID and announcement ID
course_id = "1234567890"
```

```
announcement_id = "announcement_12345"

# Define the announcement patch details
patch_data = {
    "text": "Updated announcement text with new details."
}

# Define the headers for the request, including the API key
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the endpoint URL
endpoint_url = f"{base_url}/courses/{course_id}/announcements/{announcement_id}"

try:
    # Make the PATCH request to update the announcement
    response = requests.patch(endpoint_url, json=patch_data, headers=headers)

    # Check if the request was successful
    if response.status_code == 200:
        print("Announcement updated successfully.")
    else:
        print(f"Failed to update announcement. Status code: {response.status_code}")
        print(f"Response: {response.text}")

except requests.exceptions.RequestException as e:
    # Handle any requests exceptions
    print(f"An error occurred: {e}")
```

Google API Documentation

Endpoint: courses.announcements.addOnAttachments.create

HTTP Method: POST

Path: v1/courses/{courseId}/announcements/{itemId}/addOnAttachments

Description: Creates an add-on attachment under a post. Requires the add-on to have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

API Endpoint: POST classroom.courses.announcements.addOnAttachments.create

Common Use Cases:
- Uploading attachments for a classroom announcement.
- Adding multiple files (e.g., PDFs, images) to a specific announcement.
- Notifying students about additional resources via attachments.

Example Request:
```http
POST /classroom/courses/{courseId}/announcements/{announcementId}/attachments
Content-Type: multipart/form-data
Authorization: Bearer {access_token}

Form Data:
{
  "file": [file1, file2]
}
```

Common Parameters:
- **courseId** (required): The ID of the course to which the announcement belongs.
- **announcementId** (required): The ID of the announcement to which the attachments are being added.
- **file** (required): The file(s) to be attached. Can be multiple files uploaded in a single request.
- **access_token** (required): The authentication token for the API user making the request.

Example Code

```python
import requests

def add_announcement_attachment(course_id, announcement_id, file_path, access_token):
    """
    Adds an attachment to an announcement in a Google Classroom course.

    :param course_id: The ID of the course.
    :param announcement_id: The ID of the announcement.
    :param file_path: The path to the file to be uploaded as an attachment.
```

```
    :param access_token: The OAuth 2.0 access token.
    :return: The response from the API call.
    """

    # Define the API endpoint
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments"

    # Open the file in binary mode
    with open(file_path, 'rb') as file:
        file_content = file.read()

    # Set up the headers
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'multipart/form-data'
    }

    # Set up the files payload
    files = {
        'file': (file_path, file_content, 'application/octet-stream')
    }

    try:
        # Make the API request
        response = requests.post(url, headers=headers, files=files)

        # Check for HTTP errors
        response.raise_for_status()

        # Return the JSON response
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f'HTTP error occurred: {http_err}')

    except Exception as err:
        # Handle other exceptions
        print(f'An error occurred: {err}')

    return None

# Example usage
course_id = 'your_course_id'
announcement_id = 'your_announcement_id'
file_path = 'path_to_your_file'
access_token = 'your_oauth_2_0_access_token'
```

```
response = add_announcement_attachment(course_id, announcement_id, file_path, access_token)
if response:
    print('Attachment added successfully:', response)
```

Google API Documentation

Endpoint: courses.announcements.addOnAttachments.list

HTTP Method: GET

Path: v1/courses/{courseId}/announcements/{itemId}/addOnAttachments

Description: Returns all attachments created by an add-on under the post. Requires the add-on to have active attachments on the post or have permission to create new attachments on the post. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description

Common Use Cases:
- Retrieve a list of attachments added to a specific announcement in a classroom course.
- Verify the list of attachments for a given announcement.
- Integrate with classroom management systems for comprehensive attachment tracking.

Example Request:
```http
GET /classroom.courses.announcements.addOnAttachments.list?courseId=12345&announcementId=67890
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **announcementId**: The unique identifier for the announcement.
- **pageSize**: The number of attachments to return per page. (Optional)
- **pageToken**: A token to specify the next page of results. (Optional)

Endpoint:
```
GET classroom.courses.announcements.addOnAttachments.list
```

Example Code

```python
import requests

# Define the API endpoint and parameters
url = "https://classroom.googleapis.com/v1/courses/COURSE_ID/announcements/ANNOUNCEMENT_ID/attachments"
params = {
    "key": "YOUR_API_KEY",  # Replace with your actual API key
    "fields": "id,title",   # Fields to be returned in the response
    "pageSize": 10          # Number of results to return per page
}
```

```
# Function to get the list of attachments for a specific announcement
def get_announcement_attachments():
    try:
        # Make the API request
        response = requests.get(url, params=params)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Extract and print the list of attachments
        attachments = data.get("attachments", [])
        for attachment in attachments:
            print(f"ID: {attachment['id']}, Title: {attachment['title']}")

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Error occurred: {req_err}")
    except KeyError as key_err:
        print(f"Key error: {key_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Call the function to get the list of attachments
get_announcement_attachments()
```

This script defines a function to retrieve the list of attachments for a specific announcement in a Google Classroom course. It includes error handling for HTTP errors, request exceptions, and key errors, and prints the IDs and titles of the attachments. Make sure to replace `COURSE_ID`, `ANNOUNCEMENT_ID`, and `YOUR_API_KEY` with actual values.

Google API Documentation

Endpoint: courses.announcements.addOnAttachments.patch

HTTP Method: PATCH

Path: v1/courses/{courseId}/announcements/{itemId}/addOnAttachments/{attachmentId}

Description: Updates an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: PATCH classroom.courses.announcements.addOnAttachments.patch

Common Use Cases:
- Updating the attachment of an announcement in a course.
- Modifying metadata or details of an existing attachment linked to an announcement.

Example Request:
```
PATCH /classroom/courses/12345/announcements/67890/addOnAttachments
Content-Type: application/json

{
  "fileId": "newFileId12345",
  "title": "Updated Assignment",
  "mimeType": "application/pdf",
  "size": "1024"
}
```

Common Parameters:
- **fileId** (string): The unique identifier for the new attachment file.
- **title** (string, optional): The title of the attachment.
- **mimeType** (string, optional): The MIME type of the attachment.
- **size** (number, optional): The size of the attachment in bytes.

Example Code

```python
import requests

# Define the API endpoint and necessary parameters
url = "https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments"

# Replace with actual values
course_id = "1234567890"
announcement_id = "9876543210"
access_token = "your_access_token_here"
```

```
# Define the headers for the request
headers = {
    "Authorization": f"Bearer {access_token}",
    "Content-Type": "application/json"
}

# Define the data to be sent in the PATCH request
data = {
    "attachments": [
        {
            "title": "Updated Assignment",
            "fileId": "new_file_id",
            "mimeType": "application/pdf"
        }
    ]
}

try:
    # Make the PATCH request to update the attachments
              response    =    requests.patch(url.format(course_id=course_id,    announcement_id=announcement_id),
headers=headers, json=data)

    # Check if the request was successful
    response.raise_for_status()

    # Print the response from the API
    print("Attachment updated successfully:", response.json())
except requests.exceptions.HTTPError as http_err:
    # Handle HTTP errors
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.ConnectionError as conn_err:
    # Handle connection errors
    print(f"Connection error occurred: {conn_err}")
except requests.exceptions.Timeout as timeout_err:
    # Handle timeout errors
    print(f"Timeout error occurred: {timeout_err}")
except requests.exceptions.RequestException as req_err:
    # Handle any other request exceptions
    print(f"An error occurred: {req_err}")
```

Google API Documentation

Endpoint: courses.announcements.addOnAttachments.delete

HTTP Method: DELETE

Path: v1/courses/{courseId}/announcements/{itemId}/addOnAttachments/{attachmentId}

Description: Deletes an add-on attachment. Requires the add-on to have been the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: DELETE classroom.courses.announcements.addOnAttachments.delete

Common Use Cases:
- Removing an attachment from an announcement in a classroom course.
- Deleting an attachment that is no longer relevant or has been replaced.
- Cleaning up attachments that were mistakenly added to an announcement.

Example Request:
```
DELETE /v1/classroom/courses/{courseId}/announcements/{announcementId}/addOnAttachments/{attachmentId}
```
Placeholders:
- `{courseId}`: The ID of the course.
- `{announcementId}`: The ID of the announcement.
- `{attachmentId}`: The ID of the attachment to be deleted.

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `announcementId` (string): The unique identifier for the announcement.
- `attachmentId` (string): The unique identifier for the attachment to be deleted.

Example Request with Placeholder Values:
```
DELETE /v1/classroom/courses/12345/announcements/67890/addOnAttachments/abcdef
```

Example Code

```python
import requests

def delete_course_announcement_attachment(course_id, announcement_id, attachment_id, access_token):
    # Define the API endpoint URL
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments/{attachme
nt_id}"
```

```
    # Set up the headers with the access token for authentication
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }

    # Make the DELETE request
    response = requests.delete(url, headers=headers)

    # Check if the request was successful
    if response.status_code == 204:
        print("Attachment deleted successfully.")
    else:
        # Handle errors
        error_message = response.json().get("error", {}).get("message", "Unknown error")
        print(f"Error deleting attachment: {error_message}")
        print(f"Status Code: {response.status_code}")
        print(f"Response: {response.text}")

# Example usage
course_id = "123456789012345678901"
announcement_id = "1234567890"
attachment_id = "attachment_123"
access_token = "your_access_token_here"

delete_course_announcement_attachment(course_id, announcement_id, attachment_id, access_token)
```

Google API Documentation

Endpoint: courses.announcements.addOnAttachments.get

HTTP Method: GET

Path: v1/courses/{courseId}/announcements/{itemId}/addOnAttachments/{attachmentId}

Description: Returns an add-on attachment. Requires the add-on requesting the attachment to be the original creator of the attachment. This method returns the following error codes: * `PERMISSION_DENIED` for access errors. * `INVALID_ARGUMENT` if the request is malformed. * `NOT_FOUND` if one of the identified resources does not exist.

AI-Generated Documentation

Technical Description for API Endpoint: GET classroom.courses.announcements.addOnAttachments.get

Common Use Cases:
- Retrieving metadata and attachments for announcements in a specific course.
- Verifying the presence of attachments before downloading them.
- Integrating with other systems to list attachments for announcements.

Example Request:
```
GET /classroom.courses.announcements.addOnAttachments.get?courseId=12345&announcementId=67890
```

Common Parameters:
- **courseId**: Unique identifier for the course. (Required)
- **announcementId**: Unique identifier for the announcement within the course. (Required)
- **accessToken**: Authentication token for API access. (Required)

Placeholder Values in Example:
- courseId: `12345`
- announcementId: `67890`

Example Code

```python
import requests

def get_attachments(course_id, announcement_id, access_token):
    # Define the API endpoint URL
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments"

    # Set up headers with the access token for authentication
    headers = {
        'Authorization': f'Bearer {access_token}'
    }

    try:
```

```
        # Make the GET request to fetch the attachments
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        response.raise_for_status()

        # Parse the JSON response
        attachments = response.json()

        # Return the attachments
        return attachments

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        # Handle other requests exceptions
        print(f"An error occurred: {err}")
    except Exception as e:
        # Handle any other exceptions
        print(f"An unexpected error occurred: {e}")

# Example usage:
# Replace with actual values
course_id = 'example_course_id'
announcement_id = 'example_announcement_id'
access_token = 'your_access_token_here'

attachments = get_attachments(course_id, announcement_id, access_token)
print(attachments)
```

Google API Documentation

Endpoint: userProfiles.get

HTTP Method: GET

Path: v1/userProfiles/{userId}

Description: Returns a user profile. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to access this user profile, if no profile exists with the requested ID, or for access errors.

AI-Generated Documentation

Technical Description for GET classroom.userProfiles.get

Common Use Cases:
- Retrieve user profiles for a specific classroom.
- Fetch detailed information about students or teachers in a classroom.
- Integrate user information into third-party applications or dashboards.

Example Request:
```
GET https://api.example.com/classroom/userProfiles?classroomId=12345
```

Common Parameters:
- `classroomId` (required): The unique identifier for the classroom.
- `userType` (optional): Filter users by type (e.g., student, teacher).
- `limit` (optional): The number of profiles to return per page.
- `offset` (optional): The number of profiles to skip before returning results.
- `filter` (optional): Additional filters to apply, such as specific user attributes or roles.

Example Code

```python
import requests

def get_user_profiles(user_id, access_token):
    """
    Fetches user profiles from the API.

    Parameters:
        user_id (str): The ID of the user.
        access_token (str): The access token for authentication.

    Returns:
        dict: The user profile data.
    """
    url = "https://classroom.googleapis.com/v1/users/{}".format(user_id)
    headers = {
        "Authorization": "Bearer {}".format(access_token),
```

```
        "Accept": "application/json"
    }

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.ConnectionError as conn_err:
        print(f"Connection error occurred: {conn_err}")
    except requests.exceptions.Timeout as timeout_err:
        print(f"Timeout error occurred: {timeout_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"An error occurred: {req_err}")
    return None

# Example usage:
user_id = "user123"
access_token = "your_access_token_here"
user_profile = get_user_profiles(user_id, access_token)
if user_profile:
    print("User Profile:", user_profile)
else:
    print("Failed to retrieve user profile.")
```

Google API Documentation

Endpoint: userProfiles.guardians.get

HTTP Method: GET

Path: v1/userProfiles/{studentId}/guardians/{guardianId}

Description: Returns a specific guardian. This method returns the following error codes: * `PERMISSION_DENIED` if no user that matches the provided `student_id` is visible to the requesting user, if the requesting user is not permitted to view guardian information for the student identified by the `student_id`, if guardians are not enabled for the domain in question, or for other access errors. * `INVALID_ARGUMENT` if a `student_id` is specified, but its format cannot be recognized (it is not an email address, nor a `student_id` from the API, nor the literal string `me`). * `NOT_FOUND` if the requesting user is permitted to view guardians for the requested `student_id`, but no `Guardian` record exists for that student that matches the provided `guardian_id`.

AI-Generated Documentation

Technical Description for GET classroom.userProfiles.guardians.get

Common Use Cases:
- Retrieving guardian information for a specific user profile.
- Validating guardian details for a student in a classroom.
- Extracting guardian contact information for communication purposes.

Example Request:
```
GET /classroom.userProfiles.guardians.get?userProfileId=12345
```

Common Parameters:
- **userProfileId** (required): The unique identifier of the user profile for which guardian information is being requested.
- **classroomId** (optional): The unique identifier of the classroom associated with the user profile, if applicable.
- **guardianId** (optional): The unique identifier of a specific guardian to retrieve, if multiple guardians are associated with the user profile.

Example Code

```python
import requests

def get_guardians(user_id, access_token):
    """
    Fetches guardian information for a given user.

    :param user_id: The ID of the user.
    :param access_token: The access token for authentication.
    :return: JSON response from the API.
    """
    url = "https://classroom.googleapis.com/v1/users/{}/guardians".format(user_id)
    headers = {
```

```
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")  # Handle HTTP errors
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")  # Handle other request errors
    except Exception as e:
        print(f"An unexpected error occurred: {e}")  # Handle any other exceptions

# Example usage:
user_id = "example_user_id"
access_token = "ya29.example_token"
guardians = get_guardians(user_id, access_token)
if guardians:
    print(guardians)
```

This code defines a function `get_guardians` that takes a `user_id` and an `access_token` as parameters, constructs the API request to fetch guardian information, and includes proper error handling for different types of exceptions that might occur during the API call. The function returns the JSON response if the request is successful, otherwise it prints the error message.

Google API Documentation

Endpoint: userProfiles.guardians.delete

HTTP Method: DELETE

Path: v1/userProfiles/{studentId}/guardians/{guardianId}

Description: Deletes a guardian. The guardian will no longer receive guardian notifications and the guardian will no longer be accessible via the API. This method returns the following error codes: * `PERMISSION_DENIED` if no user that matches the provided `student_id` is visible to the requesting user, if the requesting user is not permitted to manage guardians for the student identified by the `student_id`, if guardians are not enabled for the domain in question, or for other access errors. * `INVALID_ARGUMENT` if a `student_id` is specified, but its format cannot be recognized (it is not an email address, nor a `student_id` from the API). * `NOT_FOUND` if the requesting user is permitted to modify guardians for the requested `student_id`, but no `Guardian` record exists for that student with the provided `guardian_id`.

AI-Generated Documentation

API Endpoint: DELETE classroom.userProfiles.guardians.delete

Common Use Cases:
- Remove a guardian from a user profile.
- Update user profiles by removing outdated guardians.
- Ensure compliance with privacy policies by deleting guardian information.

Example Request:
```http
DELETE /classroom/userProfiles/12345/guardians/67890 HTTP/1.1
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **userProfileId (Path Parameter)**: The unique identifier for the user profile from which the guardian will be removed.
- **guardianId (Path Parameter)**: The unique identifier for the guardian to be removed from the user profile.
- **Authorization (Header)**: Bearer token for authentication.

Example Code

```python
import requests

def delete_guardian(user_id, guardian_id):
    """
    Deletes a guardian from a user's profile.

    :param user_id: The ID of the user.
    :param guardian_id: The ID of the guardian to delete.
    :return: Response from the API.
    """
```

```
    url = "https://classroom.googleapis.com/v1/users/{}/guardians/{}".format(user_id, guardian_id)
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your actual access token
    }

    try:
        response = requests.delete(url, headers=headers)
        response.raise_for_status()  # Raises HTTPError for bad responses (4xx and 5xx)

        # Successful deletion
        print("Guardian deleted successfully.")
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
        return {"error": str(http_err)}

    except Exception as err:
        # Handle other errors
        print(f"An error occurred: {err}")
        return {"error": str(err)}

# Example usage
user_id = "1234567890"  # Replace with actual user ID
guardian_id = "0987654321"  # Replace with actual guardian ID

delete_guardian(user_id, guardian_id)
```

Google API Documentation

Endpoint: userProfiles.guardians.list

HTTP Method: GET

Path: v1/userProfiles/{studentId}/guardians

Description: Returns a list of guardians that the requesting user is permitted to view, restricted to those that match the request. To list guardians for any student that the requesting user may view guardians for, use the literal character `-` for the student ID. This method returns the following error codes: * `PERMISSION_DENIED` if a `student_id` is specified, and the requesting user is not permitted to view guardian information for that student, if `"-"` is specified as the `student_id` and the user is not a domain administrator, if guardians are not enabled for the domain in question, if the `invited_email_address` filter is set by a user who is not a domain administrator, or for other access errors. * `INVALID_ARGUMENT` if a `student_id` is specified, but its format cannot be recognized (it is not an email address, nor a `student_id` from the API, nor the literal string `me`). May also be returned if an invalid `page_token` is provided. * `NOT_FOUND` if a `student_id` is specified, and its format can be recognized, but Classroom has no record of that student.

AI-Generated Documentation

Endpoint: GET classroom.userProfiles.guardians.list

Common Use Cases:
- Retrieve a list of guardians associated with a specific user profile in a classroom.
- Verify and manage guardians for students in an educational system.
- Export guardian information for administrative purposes.

Example Request:
```
GET https://api.example.com/v1/classroom/userProfiles/guardians/list
```
- **Headers:**
  - `Authorization: Bearer YOUR_ACCESS_TOKEN`
  - `Content-Type: application/json`

Common Parameters:
- **userId** (string, required): The unique identifier of the user profile whose guardians are being listed.
- **pageToken** (string, optional): A token to retrieve the next set of results in a paginated response.
- **maxResults** (integer, optional): The maximum number of results to return per page. Default is 10; maximum is 100.
- **sortOrder** (string, optional): The order in which the guardians should be sorted (e.g., "asc" for ascending, "desc" for descending).

Example Code

```python
import requests

def get_guardian_profiles(user_id, access_token):
    """
    Fetch the list of guardians for a given user.
```

```
    Parameters:
    - user_id (str): The ID of the user.
    - access_token (str): The access token for authentication.

    Returns:
    - list: A list of guardian profiles.
    """

    # Define the API endpoint and headers
    url = "https://classroom.googleapis.com/v1/users/{}/guardians".format(user_id)
    headers = {
        "Authorization": "Bearer {}".format(access_token),
        "Content-Type": "application/json"
    }

    try:
        # Make the API request
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an exception for HTTP errors

        # Parse the JSON response
        guardians = response.json().get('guardians', [])

        # Return the list of guardian profiles
        return guardians

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Request error occurred: {req_err}")
    except Exception as err:
        print(f"An error occurred: {err}")

    return []

# Example usage
if __name__ == "__main__":
    user_id = "example_user_id"
    access_token = "example_access_token"
    guardians = get_guardian_profiles(user_id, access_token)
    print(guardians)
```

Google API Documentation

Endpoint: userProfiles.guardianInvitations.patch

HTTP Method: PATCH

Path: v1/userProfiles/{studentId}/guardianInvitations/{invitationId}

Description: Modifies a guardian invitation. Currently, the only valid modification is to change the `state` from `PENDING` to `COMPLETE`. This has the effect of withdrawing the invitation. This method returns the following error codes: * `PERMISSION_DENIED` if the current user does not have permission to manage guardians, if guardians are not enabled for the domain in question or for other access errors. * `FAILED_PRECONDITION` if the guardian link is not in the `PENDING` state. * `INVALID_ARGUMENT` if the format of the student ID provided cannot be recognized (it is not an email address, nor a `user_id` from this API), or if the passed `GuardianInvitation` has a `state` other than `COMPLETE`, or if it modifies fields other than `state`. * `NOT_FOUND` if the student ID provided is a valid student ID, but Classroom has no record of that student, or if the `id` field does not refer to a guardian invitation known to Classroom.

AI-Generated Documentation

Endpoint: PATCH classroom.userProfiles.guardianInvitations.patch

Common Use Cases:
- Update the status of guardian invitations.
- Modify guardian invitation details such as email address or invitation message.

Example Request:
```json
PATCH /classroom/userProfiles/guardianInvitations/123
Content-Type: application/json

{
  "status": "accepted",
  "email": "new.email@example.com",
  "message": "Invitation updated successfully."
}
```

Common Parameters:
- **status** (string): The status of the guardian invitation (e.g., "sent", "accepted", "declined").
- **email** (string): The email address of the guardian.
- **message** (string): A custom message for the guardian invitation.

Example Code

```python
import requests

# Define the API endpoint and the necessary headers
url = "https://classroom.googleapis.com/v1/userProfiles.guardianInvitations"
headers = {
```

```
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the patch data with realistic parameter names and values
patch_data = {
    "emailAddress": "new.guardian@example.com",  # New guardian's email address
    "invitationId": "INVITATION_ID"  # The ID of the invitation to be updated
}

# Define the request payload
payload = {
    "guardianInvitations": patch_data
}

try:
    # Make the API request
    response = requests.patch(url, headers=headers, json=payload)

    # Check if the request was successful
    response.raise_for_status()

    # Print the response from the API
    print("Request was successful")
    print("Response:", response.json())

except requests.exceptions.HTTPError as http_err:
    # Handle HTTP errors
    print(f"HTTP error occurred: {http_err}")
except Exception as err:
    # Handle any other exceptions
    print(f"An error occurred: {err}")
```

Google API Documentation

Endpoint: userProfiles.guardianInvitations.list

HTTP Method: GET

Path: v1/userProfiles/{studentId}/guardianInvitations

Description: Returns a list of guardian invitations that the requesting user is permitted to view, filtered by the parameters provided. This method returns the following error codes: * `PERMISSION_DENIED` if a `student_id` is specified, and the requesting user is not permitted to view guardian invitations for that student, if `"-"` is specified as the `student_id` and the user is not a domain administrator, if guardians are not enabled for the domain in question, or for other access errors. * `INVALID_ARGUMENT` if a `student_id` is specified, but its format cannot be recognized (it is not an email address, nor a `student_id` from the API, nor the literal string `me`). May also be returned if an invalid `page_token` or `state` is provided. * `NOT_FOUND` if a `student_id` is specified, and its format can be recognized, but Classroom has no record of that student.

AI-Generated Documentation

Endpoint: GET classroom.userProfiles.guardianInvitations.list

Common Use Cases:
- Retrieve a list of guardian invitations sent to a user's classroom.
- Check the status of pending guardian invitations.
- Identify guardians who have accepted or rejected invitations.

Example Request:
```
GET
https://classroom.googleapis.com/v1/users/{userId}/classroom.userProfiles.guardianInvitations.list?pageToken={pageToken}
Authorization: Bearer {access_token}
```

Common Parameters:
- **userId** (string, required): The identifier of the user whose guardian invitations are being listed.
- **pageToken** (string, optional): The token for fetching the next page of results.
- **pageSize** (integer, optional): The maximum number of results to return per page. Default is 100, maximum is 1000.

Example Code

```python
import requests

# Define the API endpoint and required parameters
api_endpoint = "https://classroom.googleapis.com/v1/courses/me/userProfiles/guardianInvitations"

# Define headers including authorization token
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
```

```
}

# Define parameters for the request
params = {
    "courseId": "1234567890",
    "invitationId": "abcdefghijklmnopqrstuvwxyz"
}

try:
    # Make the GET request to the API
    response = requests.get(api_endpoint, headers=headers, params=params)

    # Check for HTTP errors
    response.raise_for_status()

    # Parse the JSON response
    invitations = response.json()

    # Print the response (for demonstration purposes)
    print(invitations)

except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as req_err:
    print(f"Error occurred: {req_err}")
except ValueError as val_err:
    print(f"Error decoding JSON: {val_err}")
```

Explanation:
1. **Importing Libraries**: `requests` library is imported to handle HTTP requests.
2. **API Endpoint and Headers**: The API endpoint URL and headers (including the authorization token) are defined.
3. **Parameters**: Realistic parameter names and values for `courseId` and `invitationId` are set.
4. **Making the Request**: A GET request is made to the API with the defined parameters and headers.
5. **Error Handling**: Various exceptions are caught and handled:
   - `HTTPError` for HTTP errors.
   - `RequestException` for general request errors.
   - `ValueError` for JSON decoding errors.
6. **Response Handling**: The JSON response is parsed and printed.

Google API Documentation

Endpoint: userProfiles.guardianInvitations.get

HTTP Method: GET

Path: v1/userProfiles/{studentId}/guardianInvitations/{invitationId}

Description: Returns a specific guardian invitation. This method returns the following error codes: * `PERMISSION_DENIED` if the requesting user is not permitted to view guardian invitations for the student identified by the `student_id`, if guardians are not enabled for the domain in question, or for other access errors. * `INVALID_ARGUMENT` if a `student_id` is specified, but its format cannot be recognized (it is not an email address, nor a `student_id` from the API, nor the literal string `me`). * `NOT_FOUND` if Classroom cannot find any record of the given student or `invitation_id`. May also be returned if the student exists, but the requesting user does not have access to see that student.

AI-Generated Documentation

Technical Description

Use Cases
- Retrieve a list of guardian invitations for a specific user profile.
- Monitor the status of sent guardian invitations.
- Verify the details of invitations sent to guardians.

Example Request
```http
GET /classroom.userProfiles.guardianInvitations/get?userId=12345&status=pending
```

Common Parameters
- **userId**: Unique identifier for the user profile.
- **status**: Filter invitations by status (e.g., pending, accepted, declined).
- **limit**: Maximum number of results to return (optional).
- **offset**: Number of results to skip (useful for pagination, optional).

Example Code

```python
import requests

def get_guardian_invitations(user_id, access_token):
    # Define the API endpoint URL
    url = 'https://classroom.googleapis.com/v1/users/{}/guardianInvitations'.format(user_id)

    # Set up the headers with the access token for authorization
    headers = {
        'Authorization': 'Bearer {}'.format(access_token)
    }

    try:
```

```
        # Make the GET request to the API
        response = requests.get(url, headers=headers)

        # Raise an exception if the request was unsuccessful
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Return the data
        return data

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')
    except requests.exceptions.RequestException as err:
        print(f'Error occurred: {err}')
    except Exception as err:
        print(f'An unexpected error occurred: {err}')

# Example usage:
user_id = '1234567890'  # Replace with a valid user ID
access_token = 'your_access_token_here'  # Replace with a valid access token

invitations = get_guardian_invitations(user_id, access_token)
if invitations:
    print(invitations)
```

Google API Documentation

Endpoint: userProfiles.guardianInvitations.create

HTTP Method: POST

Path: v1/userProfiles/{studentId}/guardianInvitations

Description: Creates a guardian invitation, and sends an email to the guardian asking them to confirm that they are the student's guardian. Once the guardian accepts the invitation, their `state` will change to `COMPLETED` and they will start receiving guardian notifications. A `Guardian` resource will also be created to represent the active guardian. The request object must have the `student_id` and `invited_email_address` fields set. Failing to set these fields, or setting any other fields in the request, will result in an error. This method returns the following error codes: * `PERMISSION_DENIED` if the current user does not have permission to manage guardians, if the guardian in question has already rejected too many requests for that student, if guardians are not enabled for the domain in question, or for other access errors. * `RESOURCE_EXHAUSTED` if the student or guardian has exceeded the guardian link limit. * `INVALID_ARGUMENT` if the guardian email address is not valid (for example, if it is too long), or if the format of the student ID provided cannot be recognized (it is not an email address, nor a `user_id` from this API). This error will also be returned if read-only fields are set, or if the `state` field is set to to a value other than `PENDING`. * `NOT_FOUND` if the student ID provided is a valid student ID, but Classroom has no record of that student. * `ALREADY_EXISTS` if there is already a pending guardian invitation for the student and `invited_email_address` provided, or if the provided `invited_email_address` matches the Google account of an existing `Guardian` for this user.

AI-Generated Documentation

Technical Description:

Endpoint: POST classroom.userProfiles.guardianInvitations.create

Common Use Cases:
- Invite guardians to a student's classroom.
- Manage and update guardian invitations.
- Facilitate communication between parents/guardians and educators.

Example Request:
```json
POST https://api.example.com/v1/classroom/userProfiles/guardianInvitations
Content-Type: application/json

{
  "studentId": "student123",
  "guardianEmail": "guardian@example.com",
  "inviterId": "teacher456",
  "invitationMessage": "You are invited to join your student's classroom."
}
```

Common Parameters:
- `studentId` (string): Unique identifier for the student whose guardian is being invited.
- `guardianEmail` (string): Email address of the guardian being invited.
- `inviterId` (string): Identifier for the user (typically a teacher) sending the invitation.

- `invitationMessage` (string, optional): Custom message to include with the invitation.

Example Code

```python
import requests

def create_guardian_invitation(access_token, user_id, guardian_email, guardian_name):
    """
    Create a guardian invitation for a user in the classroom API.

    Parameters:
    - access_token (str): The OAuth 2.0 access token.
    - user_id (str): The user ID of the student.
    - guardian_email (str): The email address of the guardian.
    - guardian_name (str): The name of the guardian.

    Returns:
    - dict: The response from the API.
    """
    url = "https://classroom.googleapis.com/v1/users/{}/guardianInvitations".format(user_id)
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }
    payload = {
        "guardianEmail": guardian_email,
        "guardianName": guardian_name
    }

    try:
        response = requests.post(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an HTTPError for bad responses
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as req_err:
        print(f"Error occurred: {req_err}")
    except Exception as err:
        print(f"An unexpected error occurred: {err}")

# Example usage
# access_token = "your_oauth2_access_token"
# user_id = "1234567890"
# guardian_email = "guardian@example.com"
# guardian_name = "John Doe"
# create_guardian_invitation(access_token, user_id, guardian_email, guardian_name)
```