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 POST classroom.registrations.create

Common Use Cases:
- Register a student for a specific classroom.
- Enroll a user in an online course.
- Add a participant to a virtual classroom session.

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

{
  "classroom_id": "12345",
  "user_id": "67890",
  "registration_date": "2023-10-01"
}
```

Common Parameters:
- **classroom_id** (string, required): Unique identifier for the classroom.
- **user_id** (string, required): Unique identifier for the user being registered.
- **registration_date** (string, optional): Date of the registration in YYYY-MM-DD format.
- **status** (string, optional): Current status of the registration (e.g., "enrolled", "pending").
- **additional_info** (object, optional): Additional information related to the registration (e.g., notes, special requirements).

Google API Documentation

Example Code

```python
import requests

def create_classroom_registration(access_token, user_id, course_id, registration_data):
    """
    Creates a new classroom registration using the classroom.registrations.create API endpoint.

    Parameters:
    - access_token (str): The access token for authentication.
    - user_id (int): The ID of the user to be registered.
    - course_id (int): The ID of the course to register the user for.
    - registration_data (dict): Additional registration data.

    Returns:
    - dict: The response from the API.
    """
    url = "https://api.example.com/classroom/registrations"
    headers = {
        "Authorization": f"Bearer {access_token}",
        "Content-Type": "application/json"
    }

    payload = {
        "user_id": user_id,
        "course_id": course_id,
        **registration_data
    }

    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 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
access_token = "your_access_token_here"
user_id = 12345
course_id = 67890
registration_data = {
    "start_date": "2023-10-01",
    "end_date": "2023-12-31",
    "additional_info": "Some additional information"
```

```
}

response = create_classroom_registration(access_token, user_id, course_id, registration_data)
print(response)
```

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:
- Unregister a student from a class.
- Remove a user from an ongoing or upcoming classroom session.
- Clean up stale or erroneous registrations.

Example Request:
```http
DELETE /v1/classrooms/12345/registrations/67890
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- `classroom_id` (path): The unique identifier for the classroom.
- `registration_id` (path): The unique identifier for the registration record to be deleted.
- `Authorization` (header): Bearer token for authenticating the API request.

Example Code

```python
import requests

def delete_classroom_registration(classroom_id, registration_id, api_key):
    # Define the API endpoint
    url = f"https://api.example.com/classroom/registrations/{registration_id}"

    # Set the headers with the API key for authentication
    headers = {
        'Authorization': f'Bearer {api_key}',
        'Content-Type': 'application/json'
    }

    # Set the parameters (if any)
    params = {
        'classroom_id': classroom_id
    }

    try:
```

```
        # Make the DELETE request to the API
        response = requests.delete(url, headers=headers, params=params)

        # Check if the request was successful
        if response.status_code == 200:
            print("Registration successfully deleted.")
        else:
            # Print the error message if the request was not successful
            print(f"Error: {response.status_code}")
            print(f"Response: {response.text}")

    except requests.exceptions.RequestException as e:
        # Handle any requests exceptions (e.g., network problems)
        print(f"An error occurred: {e}")

# Example usage
classroom_id = 12345
registration_id = 67890
api_key = "your_api_key_here"

delete_classroom_registration(classroom_id, registration_id, api_key)
```

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

API Endpoint: GET classroom.courses.get

Common Use Cases:
- Retrieve a list of courses enrolled by a student.
- Access course details for administrative or teaching purposes.
- Fetch course information for integration with other educational tools or dashboards.

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

Common Parameters:
- `student_id` (string): The unique identifier of the student whose courses are being retrieved.
- `page` (integer, optional): The page number for pagination, used to navigate through large sets of courses.
- `limit` (integer, optional): The number of courses to return per page, useful for limiting the amount of data returned.
- `sort_by` (string, optional): The field to sort the courses by (e.g., `name`, `start_date`).
- `sort_order` (string, optional): The order of sorting (e.g., `asc`, `desc`).

Example Code

```python
import requests

def get_course_details(course_id, api_key):
    url = "https://classroom.googleapis.com/v1/courses"

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

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

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

        # Raise an exception if the response status code is not 200 (OK)
        response.raise_for_status()

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

        # Return the course details
        return course_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 e:
        print(f"An unexpected error occurred: {e}")

# Example usage
course_id = "course12345"
api_key = "your_api_key_here"

course_details = get_course_details(course_id, api_key)
if course_details:
    print(course_details)
```

This Python script defines a function `get_course_details` that takes a `course_id` and an `api_key` as parameters. It sends a GET request to the `classroom.courses.get` API endpoint to retrieve course details. The script includes proper error handling for HTTP errors, request exceptions, and any unexpected errors. The example usage demonstrates how to call the function and print the course details.

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

PATCH classroom.courses.patch

Common Use Cases:
- Updating specific details of an existing course, such as the course name or description.
- Modifying the schedule or instructor of a course without deleting and recreating the course.

Example Request:
```http
PATCH /classroom/courses/12345
Content-Type: application/json

{
  "name": "Advanced Mathematics",
  "description": "An in-depth course on advanced mathematical concepts."
}
```

Common Parameters:
- **Course ID (Path Parameter)**: The unique identifier of the course to be updated. (e.g., `12345`).
- **name (Optional)**: The new name of the course.
- **description (Optional)**: The updated description of the course.
- **schedule (Optional)**: The new schedule for the course.
- **instructor (Optional)**: The new instructor for the course.
- **otherParameters (Optional)**: Any other specific parameters required for updating the course details.

Example Code

```python
import requests

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

```
}

# Define the course ID and patch data
course_id = "1234567890"
patch_data = {
    "name": "Updated Course Name",
    "section": "Updated Section",
    "room": "Updated Room"
}

# Define the patch request URL
url = url.format(course_id=course_id)

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

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

    # Print the response from the API
    print("Course updated successfully:", 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}")
```

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

Common Use Cases
- Update the details of an existing course in a classroom.
- Modify course name, description, or other relevant attributes.
- Adjust course settings such as start date, end date, or enrollment period.

Example Request
```json
PUT /classroom/courses/update
Content-Type: application/json

{
  "courseId": "12345",
  "name": "Mathematics 101",
  "description": "Introduction to Algebra and Calculus",
  "startDate": "2023-09-01",
  "endDate": "2023-12-15",
  "enrollmentPeriod": "2023-08-01 to 2023-08-31"
}
```

Common Parameters
- **courseId** (string): The unique identifier of the course to be updated.
- **name** (string): The new name of the course.
- **description** (string): A detailed description of the course.
- **startDate** (string): The new start date of the course in YYYY-MM-DD format.
- **endDate** (string): The new end date of the course in YYYY-MM-DD format.
- **enrollmentPeriod** (string): The new enrollment period for the course.

Example Code

```python
import requests

def update_course(course_id, course_name, section, description):
    """
    Updates a course in the classroom service.
```

```
    :param course_id: ID of the course to update
    :param course_name: New name of the course
    :param section: New section of the course
    :param description: New description of the course
    """
    url = "https://classroom.googleapis.com/v1/courses/{}".format(course_id)
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }
    payload = {
        "name": course_name,
        "section": section,
        "description": description
    }

    try:
        response = requests.put(url, headers=headers, json=payload)

        # Check for HTTP errors
        response.raise_for_status()

        # If the request is successful, no need to return anything
        print("Course updated successfully.")

    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:
update_course(
    course_id="12345",
    course_name="Advanced Python Programming",
    section="CS-202",
    description="This course covers advanced topics in Python programming."
)
```

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

DELETE classroom.courses.delete

Common Use Cases
- Deleting a course from the classroom system.
- Removing a specific course due to changes in the curriculum.
- Cleaning up outdated or obsolete courses.

Example Request
```http
DELETE /classroom/courses/12345
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters
- **course_id (Path Parameter)**: The unique identifier of the course to be deleted.
- **Authorization (Header)**: A Bearer token required for authentication.

Example Code

```python
import requests

def delete_course(course_id, access_token):
    # API endpoint URL
    url = "https://classroom.googleapis.com/v1/courses/{course_id}".format(course_id=course_id)

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

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

    # Check for HTTP errors
    if response.status_code == 204:
        print("Course deleted successfully.")
```

```
    elif response.status_code == 401:
        print("Unauthorized access. Please check your access token.")
    elif response.status_code == 403:
        print("Forbidden. You do not have permission to delete this course.")
    elif response.status_code == 404:
        print("Course not found.")
    else:
        print(f"An error occurred: {response.status_code}")
        print(response.text)

# Example usage
course_id = "1234567890"
access_token = "your_access_token_here"
delete_course(course_id, access_token)
```

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.
- Adding a course to a specific classroom or department.
- Setting up course details such as title, description, and schedule.

Example Request:
```json
{
  "classroomId": "12345",
  "courseTitle": "Introduction to Computer Science",
  "courseDescription": "An introductory course covering the basics of computer science.",
  "instructorId": "67890",
  "schedule": {
    "startDate": "2023-09-01",
    "endDate": "2023-12-15",
    "days": ["Mon", "Wed", "Fri"],
    "time": "10:00-12:00"
  }
}
```

Common Parameters:
- **classroomId**: The unique identifier for the classroom where the course will be added. (Required)
- **courseTitle**: The title of the course. (Required)
- **courseDescription**: A brief description of the course content. (Optional)
- **instructorId**: The unique identifier for the instructor assigned to the course. (Optional)
- **schedule**:
  - **startDate**: The start date of the course in YYYY-MM-DD format. (Required)
  - **endDate**: The end date of the course in YYYY-MM-DD format. (Required)
  - **days**: The days of the week the course will be held (e.g., ["Mon", "Wed", "Fri"]). (Required)
  - **time**: The time of day the course will be held (e.g., "10:00-12:00"). (Required)

Example Code

```python
import requests

# Define the API endpoint and required parameters
url = "https://classroom.googleapis.com/v1/courses"
api_key = "YOUR_API_KEY"
course_data = {
    "id": "1234567890",
    "name": "Introduction to Python",
    "section": "A",
    "ownerId": "owner@example.com",
    "room": "Room 101",
    "courseState": "PROVISIONED",
    "ALTERNATIVE_ID": "PYTHON101",
    "descriptionHeading": "Course Description",
    "description": "This course covers the basics of Python programming.",
    "creationTime": "2023-10-01T00:00:00Z",
}

# Include API key in the headers
headers = {
    "Authorization": f"Bearer {api_key}",
    "Content-Type": "application/json"
}

# Make the POST request to create the course
try:
    response = requests.post(url, json=course_data, headers=headers)
    # Check if the request was successful
    response.raise_for_status()
    # Print the response from the API
    print("Course created successfully!")
    print(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}")
```

This code defines an API endpoint and required parameters for creating a course in Google Classroom. It includes proper error handling to manage potential issues during the API call.

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:
- Fetching a list of all courses assigned to a user.
- Retrieving course details for administrative purposes.
- Automating course enrollment processes.

Example Request:
```
GET /classroom/courses.list?userId=12345&access_token=YOUR_ACCESS_TOKEN
```

Common Parameters:
- `userId`: Unique identifier for the user whose courses are to be listed.
- `access_token`: Authentication token to access the API.
- `pageToken`: Token for pagination, used to fetch subsequent pages of results.
- `pageSize`: Number of courses to be returned per page (optional).

Example Code

```python
import requests

def list_courses():
    # Define the API endpoint
    url = "https://classroom.googleapis.com/v1/courses"

    # Define the API key (use your actual API key here)
    api_key = "YOUR_API_KEY"

    # Set up the parameters
    params = {
        "pageSize": 10,  # Number of courses per page
        "courseStates": "ACTIVE",  # Only active courses
        "orderBy": "name"  # Sort by course name
    }
```

```
    try:
        # Make the API request
        response = requests.get(url, params=params, headers={"Authorization": f"Bearer {api_key}"})

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

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

        # Print the list of courses
        for course in courses.get("courses", []):
            print(f"Course ID: {course['id']}, Course Name: {course['name']}")

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

# Call the function to list courses
list_courses()
```

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:
- Update the list of students assigned to a coursework.
- Add or remove students to/from an existing coursework assignment.
- Reassign coursework to different students.

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

{
  "addStudentIds": ["student1", "student2"],
  "removeStudentIds": ["student3"]
}
```

Common Parameters:
- **courseId** (string): The ID of the course.
- **courseWorkId** (string): The ID of the coursework.
- **addStudentIds** (array of strings): A list of student IDs to be added as assignees.
- **removeStudentIds** (array of strings): A list of student IDs to be removed as assignees.

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

Technical Description

Endpoint: DELETE classroom.courses.courseWork.delete

Common Use Cases:
- Deleting a coursework assignment from a class.
- Removing outdated or incorrect coursework to maintain an up-to-date curriculum.

Example Request:
```http
DELETE https://api.example.com/classroom/courses/{courseId}/courseWork/{courseWorkId}
Authorization: Bearer {access_token}
```

Common Parameters:
- **courseId**: (Path) The unique identifier for the course to which the coursework belongs.
- **courseWorkId**: (Path) The unique identifier for the coursework to be deleted.
- **Authorization**: (Header) Bearer token for authentication.

These parameters help identify the specific course and coursework to be deleted, ensuring that the correct data is targeted for removal.

Example Code

```python
import requests

def delete_course_work(course_id, course_work_id, api_key):
    """
    Delete a course work from a specified course.

    Parameters:
    course_id (str): The ID of the course.
    course_work_id (str): The ID of the course work to be deleted.
```

```
    api_key (str): The API key for authentication.

    Returns:
    bool: True if the course work was successfully deleted, False otherwise.
    """
    # Set the API endpoint
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}"

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

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

        # Check if the request was successful
        if response.status_code == 204:
            print("Course work deleted successfully.")
            return True
        else:
            print(f"Failed to delete course work. Status code: {response.status_code}")
            print(response.json())  # Print the response JSON for debugging
            return False

    except requests.exceptions.RequestException as e:
        # Handle any request exceptions
        print(f"An error occurred: {e}")
        return False

# Example usage
course_id = "1234567890"  # Replace with a valid course ID
course_work_id = "0987654321"  # Replace with a valid course work ID
api_key = "YOUR_API_KEY"  # Replace with your actual API key

# Call the function to delete the course work
delete_course_work(course_id, course_work_id, api_key)
```

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

API Endpoint: POST classroom.courses.courseWork.create

Common Use Cases:
- Creating a new assignment for a class.
- Adding an essay or project task for students.
- Scheduling a quiz or exam within a course.

Example Request:
```json
POST https://classroom.google.com/v1/classroom.courses.courseWork.create
{
  "title": "Weekly Math Quiz",
  "description": "This quiz covers the topics discussed in the last week.",
  "courseId": "1234567890",
  "materials": [
    {
      "link": {
        "url": "https://example.com/quiz-questions",
        "title": "Quiz Questions"
      }
    }
  ],
  "dueDate": {
    "year": 2023,
    "month": 10,
    "day": 15
  },
  "maxPoints": 100
}
```

Common Parameters:

- **title** (string): The title of the coursework.
- **description** (string): A detailed description of the coursework.
- **courseId** (string): The ID of the course to which the coursework belongs.
- **materials** (array of objects): List of materials related to the coursework, each material can include a `link` with `url` and `title`.
- **dueDate** (object): The due date for the coursework, including `year`, `month`, and `day`.
- **maxPoints** (integer): The maximum points the coursework is worth.

Example Code

```python
import requests

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

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

# Define the parameters for the request
parameters = {
    "courseId": "1234567890",
    "title": "Homework Assignment",
    "description": "Complete the following problems from the text book.",
    "dueDate": {
        "year": 2023,
        "month": 12,
        "day": 15
    },
    "state": "PUBLISHED"
}

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

# Check if the request was successful
if response.status_code == 200:
    print("Course work created successfully!")
    print(response.json())
else:
    # Handle errors
    print(f"Failed to create course work. Status code: {response.status_code}")
    try:
        error_details = response.json()
        print(f"Error details: {error_details}")
    except ValueError:
```

```
        print("Failed to parse error details.")
```

This code demonstrates how to interact with the `classroom.courses.courseWork.create` API endpoint to create a course work assignment. It includes proper error handling and prints the response or error details accordingly.

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

Technical Description for GET classroom.courses.courseWork.getAddOnContext

Common Use Cases
1. **Retrieve Contextual Information**: Use this endpoint to fetch contextual information for a specific coursework item, such as due date, assignment details, and other relevant data.
2. **Enhance User Experience**: Implement this in educational applications to provide a seamless user experience by displaying detailed coursework information.
3. **Integration with Third-Party Tools**: Use this endpoint to integrate with third-party educational tools that need detailed coursework data.

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

Common Parameters
- **courseId**: (string, required) The ID of the course.
- **courseWorkId**: (string, required) The ID of the coursework item.

Example Code

```python
import requests

def get_add_on_context(course_id, user_id, add_on_id):
    """
    Fetch the add-on context for a specific course and user.

    :param course_id: The ID of the course.
    :param user_id: The ID of the user.
    :param add_on_id: The ID of the add-on.
    :return: The JSON response from the API.
    """
        url = "https://classroom.googleapis.com/v1/courses/" + course_id + "/courseWork/" + add_on_id +
```

```
"/getAddOnContext"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }

    # Prepare query parameters
    params = {
        "userId": user_id
    }

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

        # 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}")  # Print HTTP errors
    except Exception as err:
        print(f"Other error occurred: {err}")  # Print other errors

# Example usage
course_id = "1234567890"
user_id = "0987654321"
add_on_id = "add-on-id-123"

# Fetch the add-on context
add_on_context = get_add_on_context(course_id, user_id, add_on_id)
print(add_on_context)
```

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

Endpoint: GET classroom.courses.courseWork.get

Common Use Cases:
- Retrieve a specific coursework item.
- Check the status and details of an assignment.
- Fetch coursework for review or grading.

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

Common Parameters:
- `courseId` (string): The ID of the course.
- `id` (string): The ID of the coursework item.
- `courseState` (string, optional): The state of the course. Acceptable values are "PROVISIONED", "ARCHIVE".
- `fields` (string, optional): Selector specifying which fields to include in a partial response.
- `quotaUser` (string, optional): An opaque string that represents a user for quota purposes. Can be any arbitrary string assigned to a user, but should not exceed 40 characters.

Example Code

```python
import requests

def get_course_work(course_id, api_key):
    """
    Function to get course work for a specific course using the classroom.courses.courseWork.get API.
    """
    # Define the endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork"

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

```
    }

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

        # Check if the request was successful (status code 200)
        if response.status_code == 200:
            # Return the JSON response
            return response.json()
        else:
            # Return an error message if the request failed
            return {"error": f"Failed to retrieve course work. Status code: {response.status_code}"}

    except requests.exceptions.RequestException as e:
        # Handle any request exceptions (e.g., network issues)
        return {"error": f"An error occurred: {str(e)}"}

# Example usage
course_id = "1234567890"  # Replace with a real course ID
api_key = "YOUR_API_KEY_HERE"  # Replace with a real API key

course_work = get_course_work(course_id, api_key)
print(course_work)
```

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 PATCH classroom.courses.courseWork.patch

Common Use Cases:
- Update the details of an existing coursework assignment.
- Modify due dates, titles, or descriptions of coursework.

Example Request:
```http
PATCH https://api.example.com/classroom.courses/123456789/courseWork/987654321
Content-Type: application/json

{
  "title": "New Assignment Title",
  "dueDate": "2024-12-31T23:59:59Z",
  "description": "Updated description for the assignment."
}
```

Common Parameters:
- `courseId` (Path Parameter): The identifier for the course.
- `courseWorkId` (Path Parameter): The identifier for the coursework assignment.
- `title` (Request Body): The new title for the coursework assignment.
- `dueDate` (Request Body): The new due date for the coursework assignment in RFC 3339 format.
- `description` (Request Body): The updated description for the coursework assignment.

Example Code

```python
import requests

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

```
# Define the parameters
course_id = "your_course_id"  # Replace with your course ID
course_work_id = "your_course_work_id"  # Replace with your course work ID
access_token = "your_access_token"  # Replace with your access token

# Define the updated course work data
updated_data = {
    "title": "Updated Course Work Title",
    "description": "Updated description for the course work.",
    "maxPoints": 100
}

# Set up the headers with the access token
headers = {
    "Authorization": f"Bearer {access_token}",
    "Content-Type": "application/json"
}

# Make the PATCH request
response = requests.patch(
    url.format(courseId=course_id, courseWorkId=course_work_id),
    headers=headers,
    json=updated_data
)

# Check for HTTP errors
if response.status_code == 200:
    print("Course work updated successfully.")
    print(response.json())
elif response.status_code == 400:
    print("Bad Request: Ensure all parameters are correct.")
elif response.status_code == 401:
    print("Unauthorized: Check your access token.")
elif response.status_code == 403:
    print("Forbidden: You may not have permission to update this course work.")
elif response.status_code == 404:
    print("Not Found: The course or course work may not exist.")
else:
    print(f"An error occurred: {response.status_code}")
    print(response.text)
```

Google API Documentation

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

Endpoint: PATCH classroom.courses.courseWork.updateRubric

Common Use Cases:
1. Updating the rubric criteria for a specific coursework assignment.
2. Modifying the scoring guidelines or assessment criteria dynamically.
3. Adjusting the rubric to align with changes in teaching methodology or curriculum.

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

{
  "id": "example_id",
  "title": "Updated Rubric",
  "description": "Updated criteria for grading",
  "pointsPossible": 100,
  "criteria": [
    {
      "description": "Updated criterion 1",
      "points": 30
    },
    {
      "description": "Updated criterion 2",
      "points": 70
    }
  ]
}
```

```

**Common Parameters:**
1. `courseId` (string): The ID of the course.
2. `courseWorkId` (string): The ID of the coursework.
3. `id` (string): The ID of the rubric.
4. `title` (string): The title of the rubric.
5. `description` (string): The description of the rubric.
6. `pointsPossible` (integer): The total points possible for the rubric.
7. `criteria` (array): An array of rubric criteria, each containing:
   - `description` (string): Description of the criterion.
   - `points` (integer): Points allocated for the criterion.

## Example Code

```python
import requests

def update_coursework_rubric(course_id, coursework_id, rubric_id, title, description):
    # Define the API endpoint and headers
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{coursework_id}/updateRubric"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }

    # Define the request payload with realistic parameter names and values
    payload = {
        "title": title,
        "description": description,
        "rubricId": rubric_id
    }

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

        # Check for successful response
        if response.status_code == 200:
            return response.json()
        else:
            # Handle different error cases
            if response.status_code == 400:
                print("Bad Request: Invalid input parameters or missing required fields.")
            elif response.status_code == 401:
                print("Unauthorized: Invalid or missing access token.")
            elif response.status_code == 403:
                print("Forbidden: Access denied for the requested operation.")
            elif response.status_code == 404:
```
```

```
                print("Not Found: The specified course or coursework was not found.")
            else:
                print(f"An error occurred: {response.status_code} - {response.text}")
            return None

    except requests.exceptions.RequestException as e:
        # Handle network-related errors
        print(f"Network error: {e}")
        return None

# Example usage
course_id = "your_course_id"
coursework_id = "your_coursework_id"
rubric_id = "your_rubric_id"
title = "Updated Rubric Title"
description = "Updated rubric description for the coursework."

updated_rubric = update_coursework_rubric(course_id, coursework_id, rubric_id, title, description)
if updated_rubric:
    print("Rubric updated successfully:", updated_rubric)
else:
    print("Failed to update rubric.")
```

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

Technical Description for GET classroom.courses.courseWork.list

Common Use Cases:
1. Retrieve a list of coursework items for a specific course.
2. Monitor assignments, quizzes, and other coursework for a class.
3. Track student submissions and deadlines.

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

Common Parameters:
1. **courseId**: The ID of the course for which to list the coursework.
2. **access_token**: OAuth 2.0 token to authenticate the request.
3. **pageToken** (optional): Token to retrieve the next set of results.
4. **maxResults** (optional): Maximum number of results to return.

Example Code

```python
import requests

def list_course_work(course_id, access_token):
    # Define the API endpoint
    url = "https://classroom.googleapis.com/v1/courses/{}/courseWork".format(course_id)

    # Define headers with the authorization token
    headers = {
        "Authorization": "Bearer {}".format(access_token)
    }

    # Define parameters for the API request (optional, can be modified as needed)
    params = {
```

```
        "pageSize": 10,  # Number of results to return per page
        "courseState": "ACTIVE"  # Filter courses by their state
    }

    try:
        # Make the GET request to the API
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()  # Raise an exception for 4xx/5xx status codes

        # Parse the JSON response
        course_work = response.json()
        return course_work

    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 ValueError as json_err:
        print(f"JSON decode error: {json_err}")

# Example usage
course_id = "1234567890"
access_token = "your_access_token_here"
course_work_list = list_course_work(course_id, access_token)
print(course_work_list)
```

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

Common Use Cases:
- Update the status of a student submission.
- Modify submission details such as feedback or grades.
- Correct metadata associated with a student submission.

Example Request:
```http
PATCH /v1/classroom/courses/{courseId}/courseWork/{courseWorkId}/studentSubmissions/{submissionId}
Content-Type: application/json

{
  "status": "graded",
  "feedback": "Great job!",
  "grade": "A"
}
```

Common Parameters:
- **courseId** (Path Parameter): The unique identifier for the course.
- **courseWorkId** (Path Parameter): The unique identifier for the coursework.
- **submissionId** (Path Parameter): The unique identifier for the student submission.
- **status** (Optional Body Parameter): The current status of the submission (e.g., "graded", "pending").
- **feedback** (Optional Body Parameter): Feedback provided to the student.
- **grade** (Optional Body Parameter): The grade assigned to the submission.

Example Code

```python
import requests

def patch_student_submission(course_id, assignment_id, student_id, access_token, new_grade):
```

```
                                                                                                    url                     =
f'https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{assignment_id}/studentSubmissions/{studen
t_id}:patch'

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

    payload = {
        'assignedGrade': new_grade
    }

    try:
        response = requests.patch(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an error for bad status codes
        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 Exception as err:
        print(f'Other error occurred: {err}')

# Example usage:
course_id = '1234567890'
assignment_id = '0987654321'
student_id = '1234567890abcdef'
access_token = 'your_access_token_here'
new_grade = '85'

result = patch_student_submission(course_id, assignment_id, student_id, access_token, new_grade)
print(result)
```

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

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

Common Use Cases:
- Retrieve a list of student submissions for a specific assignment in a class.
- Fetch submission details for monitoring student progress.
- Review and grade student assignments.

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

Common Parameters:
- **courseId** (Path): The ID of the course.
- **courseWorkId** (Path): The ID of the coursework.
- **pageToken** (Query): Token to specify a page for large result sets.
- **studentId** (Query): The ID of the student to filter submissions.
- **returnRevisions** (Query): Whether to return a list of revisions for each submission.

Example Code

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

# Function to get student submissions for a course work item
def get_student_submissions(course_id, assignment_id, student_id):
    # Authenticate and build the service
    service = build('classroom', 'v1', credentials=google_auth.get_credentials())

    try:
        # Make the API request to get student submissions
        results = service.courses().courseWork().studentSubmissions().get(
            courseId=course_id,
            courseWorkId=assignment_id,
            studentId=student_id
        ).execute()
```

```
        # Return the submission details
        return results

    except google_auth.exceptions.HttpError as error:
        # Handle HTTP errors
        print(f'An error occurred: {error}')
        return None

# Example usage
course_id = '1234567890'  # Replace with a real course ID
assignment_id = 'abcdefghij'  # Replace with a real assignment ID
student_id = 'student123'  # Replace with a real student ID

submission = get_student_submissions(course_id, assignment_id, student_id)
if submission:
    print('Student Submission:', submission)
else:
    print('Failed to retrieve student submission')
```

This script authenticates with Google Classroom, makes an API request to get student submissions for a specific course work item, and handles potential errors that may occur during the API call.

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

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

Common Use Cases:
1. **Returning a submission:** Teachers or administrators can use this endpoint to return a submission to a student for further work.
2. **Reassessment:** Instructors might return a submission to reassess a student after revisions.
3. **Feedback Implementation:** Students can resubmit their work after addressing feedback provided by the teacher.

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

{
  "assignmentId": "12345",
  "studentId": "student@example.com",
  "assignmentId": "67890",
  "text": "Please resubmit your assignment with corrections."
}
```

Common Parameters:
- **courseId (Path Parameter):** The unique identifier for the course.
- **courseWorkId (Path Parameter):** The unique identifier for the coursework submission.
- **studentId (Path Parameter):** The unique identifier for the student.
- **text (Optional Body Parameter):** Additional feedback or instructions for the student.
- **assignmentId (Optional Body Parameter):** The unique identifier for the specific assignment.

Google API Documentation

Example Code

```python
import requests

# Define the API endpoint and parameters
api_endpoint = "https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions"
params = {
    "courseId": "1234567890",  # Replace with the actual course ID
    "courseWorkId": "9876543210",  # Replace with the actual course work ID
    "alt": "json"  # Specify the response format
}

# Define the headers for the API request
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your actual access token
}

try:
    # Make the API request
    response = requests.get(api_endpoint, params=params, 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.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}")
```

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

Technical Description for API Endpoint: POST classroom.courses.courseWork.studentSubmissions.turnIn

Common Use Cases:
1. **Submitting Assignments:**
   Students can submit their completed assignments or coursework to their teachers.

2. **Late Submissions:**
   Allows students to submit assignments even after the due date, if permitted by the teacher.

3. **Finalizing Coursework:**
   Ensures that the submission is final and cannot be edited further.

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

{
  "courseWorkId": "COURSEWORK_ID",
  "courseId": "COURSE_ID",
  "studentId": "STUDENT_ID",
  "submission": {
    "assignmentSubmission": {
      "attachments": [{
        "fileId": "FILE_ID"
      }],
      "comments": "Here are my thoughts on this assignment."
    }
  }
}
```

Common Parameters:
- **courseWorkId (string):**
  The ID of the coursework to be turned in.

- **courseId (string):**
  The ID of the course to which the submission belongs.

- **studentId (string):**
  The ID of the student submitting the coursework.

- **submission (object):**
  An object containing details about the submission, including attachments and comments.

Example Code

```python
import requests

def turn_in_coursework(course_id, course_work_id, access_token, submission):
    """
    This function sends a request to turn in student coursework.

    Args:
    course_id (str): The ID of the course.
    course_work_id (str): The ID of the coursework.
    access_token (str): The access token for authentication.
    submission (dict): The submission to be turned in.

    Returns:
    dict: The response from the API.
    """
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions"

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

    try:
        response = requests.post(url, headers=headers, json=submission)
        response.raise_for_status()  # Raises 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 specific HTTP errors if needed
    except requests.exceptions.RequestException as err:
        print(f'Error occurred: {err}')
```

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

# Example usage
course_id = '1234567890'
course_work_id = '0987654321'
access_token = 'your_access_token_here'
submission = {
    "submission": {
        "assignmentSubmission": {
            "submissionStatus": "TURNED_IN",
            "attachments": [
                {
                    "title": "Homework Assignment",
                    "description": "This is the submitted homework",
                    "fileId": "1234567890abcdefghijklmnopqrstuvwxyz"
                }
            ]
        }
    }
}

result = turn_in_coursework(course_id, course_work_id, access_token, submission)
if result:
    print("Submission successful:", result)
```

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

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

Common Use Cases:
- Retrieve a list of student submissions for a specific coursework assignment.
- Monitor student progress and ensure all students have submitted their assignments.
- Provide insights into submission timelines and completion rates for coursework.

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

Common Parameters:
- `courseId` (required): The ID of the course.
- `courseWorkId` (required): The ID of the coursework.
- `pageSize` (optional): The maximum number of submissions to return per page.
- `pageToken` (optional): The nextPageToken value returned from a previous request to retrieve the next page of results.

Example Code

```python
import requests

def list_student_submissions(course_id, assignment_id, access_token):
    """
    Function to list student submissions for a specific course and assignment.

    :param course_id: ID of the course
    :param assignment_id: ID of the assignment
    :param access_token: Access token for authentication
    :return: List of student submissions or error message
    """
    # API endpoint URL
```

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

    # Headers including the access token for authorization
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }

    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 into a Python dictionary
        submissions = response.json()

        # Return the list of student submissions
        return submissions

    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        return f"HTTP error occurred: {http_err}"
    except Exception as err:
        # Handle other exceptions
        return f"An error occurred: {err}"

# Example usage
course_id = '1234567890'
assignment_id = '0987654321'
access_token = 'YOUR_ACCESS_TOKEN'

submissions = list_student_submissions(course_id, assignment_id, access_token)
print(submissions)
```

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

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

Common Use Cases:
- Updating or replacing an attachment for a coursework submission.
- Adding new attachments to an existing submission.
- Removing attachments from a submission.

Example Request:
```json
{
  "courseId": "12345",
  "courseWorkId": "67890",
  "studentId": "abcdef",
  "submissionId": "ghijk",
  "attachments": [
    {
      "id": "lmnop",
      "title": "Updated Assignment",
      "driveFile": {
        "id": "qrstu",
        "title": "Updated Assignment",
        "mimeType": "application/pdf"
      }
    },
    {
      "id": "vwxyz",
      "title": "New Attachment",
      "driveFile": {
        "id": "abcde",
        "title": "New Attachment",
        "mimeType": "image/jpeg"
      }
    }
```

```
 ]
}
```

Common Parameters:
- `courseId`: The ID of the course.
- `courseWorkId`: The ID of the coursework.
- `studentId`: The ID of the student.
- `submissionId`: The ID of the submission.
- `attachments`: An array of attachment objects, each containing:
  - `id`: The ID of the attachment.
  - `title`: The title of the attachment.
  - `driveFile`: An object containing:
    - `id`: The ID of the Drive file.
    - `title`: The title of the Drive file.
    - `mimeType`: The MIME type of the Drive file.

Example Code

```python
import requests

def modify_coursework_submission_attachments(course_id, course_work_id, submission_id, file_id, new_content):
    # Define the API endpoint
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/{submi
ssion_id}/modifyAttachments"

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

    # Define the payload with the new content for the attachment
    payload = {
        'addAttachments': [
            {
                'title': 'New File Title',
                'fileId': file_id,  # ID of the file to be added
                'content': new_content  # New content for the attachment
            }
        ],
        'removeAttachmentIds': []  # No removal in this example
    }

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

```
        # Check for errors in the response
        if response.status_code == 200:
            return response.json()  # Return the JSON response
        else:
            # Handle different HTTP status codes
            response.raise_for_status()
    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 = '9876543210'
submission_id = '0123456789'
file_id = 'newfileid'
new_content = 'This is the new content for the attachment.'

modify_coursework_submission_attachments(course_id, course_work_id, submission_id, file_id, new_content)
```

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

Common Use Cases:
- Reclaiming student submissions after a teacher has claimed them.
- Resetting submission states to allow further actions or modifications.
- Managing re-submission processes in an educational platform.

Example Request:
```json
POST https://api.example.com/v1/classroom.courses.courseWork.studentSubmissions.reclaim
Content-Type: application/json

{
  "courseId": "12345",
  "courseWorkId": "67890",
  "studentId": "abcde",
  "submissionId": "fghij"
}
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **courseWorkId**: The unique identifier for the coursework within the course.
- **studentId**: The unique identifier for the student.
- **submissionId**: The unique identifier for the student's submission to reclaim.

Example Code

```python
import requests

def reclaim_student_submission(course_id, course_work_id, student_id, access_token):
    # Define the API endpoint
```

```
                                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/{stude
nt_id}:reclaim"

    # Set up the headers with the access token
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }

    try:
        # Make the request to reclaim the student submission
        response = requests.post(url, headers=headers)

        # Check for HTTP errors
        response.raise_for_status()

        # Return the response JSON
        return response.json()
    except requests.exceptions.HTTPError as http_err:
        # Handle HTTP errors
        print(f'HTTP error occurred: {http_err}')
        return None
    except Exception as err:
        # Handle other errors
        print(f'An error occurred: {err}')
        return None

# Example usage
course_id = '1234567890'
course_work_id = '9876543210'
student_id = 'student123'
access_token = 'your_access_token_here'

# Call the function and print the result
result = reclaim_student_submission(course_id, course_work_id, student_id, access_token)
if result:
    print('Reclamation successful:', result)
else:
    print('Reclamation failed')
```

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

Endpoint: DELETE classroom.courses.courseWork.rubrics.delete

Common Use Cases:
- Remove a specific rubric from a coursework assignment.
- Delete outdated or incorrect rubrics.
- Clean up course materials by removing unused rubrics.

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

Common Parameters:
- `courseId`: The unique identifier for the course. (Path Parameter)
- `courseWorkId`: The unique identifier for the coursework. (Path Parameter)
- `rubricId`: The unique identifier for the rubric. (Path Parameter)

Example Code

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

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

# Load credentials from the service account file
credentials = service_account.Credentials.from_service_account_file(
    SERVICE_ACCOUNT_FILE, scopes=SCOPES)
```

```
# Build the Classroom API service
service = build('classroom', 'v1', credentials=credentials)

def delete_course_work_rubric(course_id, course_work_id):
    try:
        # Attempt to delete the rubric associated with the course work
        request = service.courses().courseWork().rubrics().delete(
            courseId=course_id,
            courseWorkId=course_work_id
        )
        response = request.execute()
        print(f"Rubric deleted successfully for course work ID: {course_work_id}")
        return response

    except google.auth.exceptions.GoogleAuthError as e:
        print(f"Authorization error: {e}")
    except googleapiclient.errors.HttpError as e:
        if e.resp.status == 404:
            print(f"Rubric not found for course work ID: {course_work_id}")
        else:
            print(f"HTTP error: {e}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

# Example usage
course_id = '1234567890'  # Replace with a valid course ID
course_work_id = '0987654321'  # Replace with a valid course work ID
delete_course_work_rubric(course_id, course_work_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 coursework in a classroom.
- Validate the rubrics used for grading assignments.
- Monitor and report on grading criteria for coursework.

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

Common Parameters:
- `courseId` (required): The ID of the course.
- `courseWorkId` (required): The ID of the coursework.
- `pageSize` (optional): Maximum number of rubrics to return per page. Default is 100.
- `pageToken` (optional): Token for requesting subsequent pages of rubrics.
- `fields` (optional): Selector specifying which fields to include in a partial response.

Example Code

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

# Define the necessary scopes and credentials
SCOPES = ['https://www.googleapis.com/auth/classroom.coursework.students']
SERVICE_ACCOUNT_FILE = 'path/to/your/service-account-file.json'

# Authenticate and build the service
credentials = Credentials.from_service_account_file(SERVICE_ACCOUNT_FILE, scopes=SCOPES)
service = build('classroom', 'v1', credentials=credentials)

def get_course_work_rubrics(course_id):
    """
    Fetches the list of rubrics for a specific course work.
```
```

```
    Args:
    course_id (str): The ID of the course.

    Returns:
    dict: The list of rubrics for the course work.
    """
    try:
        # Define the parameters for the API request
        params = {
            'courseId': course_id
        }

        # Make the API request
        results = service.courses().courseWork().rubrics().list(courseId=course_id).execute()

        # Return the list of rubrics
        return results.get('rubrics', [])

    except Exception as e:
        # Handle any errors that occur during the API request
        print(f"An error occurred: {e}")
        return []

# Example usage
course_id = 'your_course_id_here'
rubrics = get_course_work_rubrics(course_id)

# Print the rubrics
print(rubrics)
```

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

API Endpoint: POST classroom.courses.courseWork.rubrics.create

Common Use Cases:
- Creating a new rubric for a specific course work item.
- Defining assessment criteria for assignments or projects.
- Structuring grading guidelines for teachers and students.

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

{
  "title": "Project Rubric",
  "description": "Criteria for grading student projects",
  "criteria": [
    {
      "title": "Creativity",
      "maxPoints": 10,
      "description": "Assess the originality and innovation of the project."
    },
    {
      "title": "Execution",
      "maxPoints": 15,
      "description": "Evaluate the technical execution and implementation."
    }
  ]
}
```

Common Parameters:
- **courseId** (Path): The ID of the course.
- **courseWorkId** (Path): The ID of the course work item.
- **title** (Body): The title of the rubric.
- **description** (Body): A brief description of the rubric.
- **criteria** (Body): An array of criteria, each containing:
  - **title**: The title of the criterion.
  - **maxPoints**: The maximum points for the criterion.
  - **description**: A description of the criterion.

Example Code

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

# Define the scope and credentials
SCOPES = ['https://www.googleapis.com/auth/classroom.courses']
SERVICE_ACCOUNT_FILE = 'path/to/service_account.json'

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

# Build the service
service = build('classroom', 'v1', credentials=credentials)

def create_rubric(course_id, title, description, points):
    try:
        # Define the rubric creation request
        rubric_request = {
            'title': title,
            'description': description,
            'points': points
        }

        # Execute the create rubric request
        rubric = service.courses().courseWork().rubrics().create(
            courseId=course_id,
            body=rubric_request
        ).execute()

        print(f"Rubric created successfully: {rubric}")
        return rubric

    except google.auth.exceptions.DefaultCredentialsError as e:
        print(f"Authentication error: {e}")
```

```
    except googleapiclient.errors.HttpError as e:
        print(f"HTTP error: {e}")
    except Exception as e:
        print(f"An error occurred: {e}")

# Example usage
course_id = 'course_id_example'
title = 'Sample Rubric'
description = 'This is a sample rubric for a course.'
points = 100  # Example points

create_rubric(course_id, title, description, points)
```

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 for GET classroom.courses.courseWork.rubrics.get

Common Use Cases:
- Fetching the rubric details for a specific piece of coursework in a classroom.
- Retrieving rubric criteria and points distribution for grading purposes.
- Integrating rubric data into external educational tools or dashboards.

Example Request:
```
GET https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/rubrics
Authorization: Bearer ya29.AHES6ZT3...
```

Common Parameters:
- `courseId` (string): The unique identifier of the course.
- `courseWorkId` (string): The unique identifier of the coursework.
- `fields` (string, optional): Selector specifying which fields to include in a partial response.
- `quotaUser` (string, optional): Available to use for quota purposes for server-side applications. Can be any arbitrary string assigned to a user, but should not exceed 40 characters. Overrides userIp if both are provided.

Example Code

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

def get_course_work_rubrics(course_id, course_work_id):
    # Define the path to your service account key file
    SERVICE_ACCOUNT_FILE = 'path/to/your/service-account-file.json'

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

    # Authenticate and create the API client
    credentials = google.oauth2.service_account.Credentials.from_service_account_file(
```

```
        SERVICE_ACCOUNT_FILE, scopes=SCOPES)

    service = build('classroom', 'v1', credentials=credentials)

    try:
        # Call the API to get the course work rubrics
        response = service.courses().courseWork().rubrics().get(
            courseId=course_id,
            courseWorkId=course_work_id
        ).execute()

        # Print the response for debugging purposes
        print(response)

        return response

    except google.auth.exceptions.DefaultCredentialsError as e:
        print(f"DefaultCredentialsError: {e}")
    except google.auth.exceptions.RefreshError as e:
        print(f"RefreshError: {e}")
    except googleapiclient.errors.HttpError as e:
        print(f"HttpError: {e}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

# Example usage
course_id = '1234567890'
course_work_id = '9876543210'
get_course_work_rubrics(course_id, course_work_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:
- Modifying specific criteria within a rubric for a course assignment.
- Updating the scoring criteria or points for different rubric categories.
- Adjusting the rubric to align with new grading standards or assignment requirements.

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

{
  "rubricId": "12345",
  "courseWorkId": "67890",
  "updatedCriteria": [
    {
      "id": "criterion1",
      "title": "New Criterion Title",
      "points": 10,
      "description": "Updated description"
    }
  ]
}
```

Common Parameters:
- **courseId** (string): The ID of the course.
- **courseWorkId** (string): The ID of the course work (assignment).

- **rubricId** (string): The ID of the rubric to be updated.
- **updatedCriteria** (array of objects): An array containing the criteria to be updated within the rubric.
  - **id** (string): The ID of the criterion to be updated.
  - **title** (string): The updated title of the criterion.
  - **points** (number): The updated points for the criterion.
  - **description** (string): The updated description of the criterion.

Example Code

```python
import requests

# Define the API endpoint and the necessary parameters
url = 'https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{courseWork_id}/rubrics'
params = {
    'course_id': '1234567890',
    'courseWork_id': '0987654321',
    'updateMask': 'description,title'
}

# Define the data to be patched
data = {
    'description': 'Newly updated rubric description',
    'title': 'Updated Rubric Title'
}

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

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

# Check for HTTP errors
if response.status_code == 200:
    print('Rubric updated successfully.')
    print(response.json())
elif response.status_code == 404:
    print('Course or course work not found.')
elif response.status_code == 401:
    print('Unauthorized access.')
else:
    print(f'Error: {response.status_code}')
    print(response.json())
```

In this script:

Google API Documentation

```
- We define the URL for the `classroom.courses.courseWork.rubrics.patch` endpoint.
- We set up the necessary parameters including `course_id`, `courseWork_id`, and `updateMask`.
- We specify the data to be patched, which includes a new description and title for the rubric.
- We include the authorization header with a Bearer token for authentication.
- We make the patch request using `requests.patch`.
- We handle different HTTP response status codes to ensure proper error handling and provide feedback.
```

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

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

Use Cases:
1. Update the metadata of attachments for coursework.
2. Modify the visibility or permissions settings of attachments.
3. Correct errors or inconsistencies in the attachment data.

Example Request:
```json
PATCH /classroom/courses/{courseId}/courseWork/{courseWorkId}/addOnAttachments/{attachmentId}
Content-Type: application/json

{
  "title": "Updated Assignment",
  "visibility": "Visible",
  "permissions": {
    "edit": true,
    "delete": false
  }
}
```

Common Parameters:
- `courseId` (Path Parameter): The unique identifier for the course.
- `courseWorkId` (Path Parameter): The unique identifier for the coursework.
- `attachmentId` (Path Parameter): The unique identifier for the attachment to be updated.
- `title` (Optional): The new title for the attachment.
- `visibility` (Optional): The visibility setting for the attachment (e.g., "Visible", "Hidden").
- `permissions` (Optional, Object): A set of permissions for the attachment, which may include:
  - `edit` (Boolean): Whether the attachment can be edited.
  - `delete` (Boolean): Whether the attachment can be deleted.

Example Code

```python
import requests
import json
```

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

# Define the parameters
courseId = "1234567890"
courseWorkId = "9876543210"
attachmentId = "0123456789"
newFileId = "9876543210"

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

# Payload for the request
payload = {
    "addOnAttachments": [
        {
            "attachmentId": attachmentId,
            "fileId": newFileId
        }
    ]
}

# Make the request
response = requests.patch(url, headers=headers, data=json.dumps(payload))

# Check for successful response
if response.status_code == 200:
    print("Successfully updated the attachments.")
    print("Response:", response.json())
else:
    # Print the error message
    print("Error:", response.status_code)
    print("Response:", response.json())
```

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

Common Use Cases:
1. Retrieve attachments added to specific coursework for review.
2. Validate the completeness of student submissions by checking attached files.
3. Facilitate grading by allowing instructors to access all submitted documents in one place.

Example Request:
```http
GET /classroom.courses.courseWork.addOnAttachments.get?courseId=1234567890&courseWorkId=9876543210
```

Common Parameters:
1. `courseId` - (Required) The unique identifier for the course.
2. `courseWorkId` - (Required) The unique identifier for the specific coursework assignment.
3. `studentId` - (Optional) The unique identifier for a specific student, if you want to retrieve attachments for a particular student.

Example Code

```python
import requests

def get_course_work_attachments(course_id, course_work_id, access_token):
    """
    Fetch course work attachments from the classroom API.

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

    Returns:
    - dict: JSON response from the API containing attachments.
    """

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

```
    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)

        # Parse the JSON response
        return response.json()

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
        return None
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
        return None

# Example usage
course_id = "1234567890"
course_work_id = "0987654321"
access_token = "your_oauth_2_access_token_here"

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

This code defines a function `get_course_work_attachments` that takes in the course ID, course work ID, and an OAuth 2.0 access token, then makes a GET request to the specified API endpoint. Proper error handling is included to manage HTTP and other request-related exceptions. The function returns the JSON response containing the attachments if successful, or `None` if an error occurs.

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

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

Common Use Cases
- Attach additional files (e.g., PDFs, images, documents) to specific coursework for students to access.
- Allow teachers to upload supplementary materials related to a coursework assignment.

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

{
  "title": "Sample Assignment Attachment",
  "id": "FILE_ID",
  "mimeType": "application/pdf",
  "size": "123456",
  "url": "https://drive.google.com/file/d/{fileId}/view?usp=sharing"
}
```

Common Parameters
1. **title (string)**: The title of the attachment.
2. **id (string)**: The unique identifier for the attachment.
3. **mimeType (string)**: The MIME type of the attachment (e.g., `application/pdf`, `image/jpeg`).
4. **size (string)**: The size of the attachment in bytes.
5. **url (string)**: The URL where the attachment can be accessed.

Example Code

```python
import requests

def add_coursework_attachment(course_id, coursework_id, file_id, access_token):
    """
    Adds an attachment to a coursework.
```

```
    :param course_id: ID of the course
    :param coursework_id: ID of the coursework
    :param file_id: ID of the file to attach
    :param access_token: OAuth 2.0 access token
    :return: Response from the API
    """

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

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

    payload = {
        "fileId": file_id,
        "title": "Sample Assignment Document",
        "driveFile": {
            "id": file_id,
            "title": "Example Document"
        }
    }

    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'
coursework_id = 'coursework_id_example'
file_id = 'file_id_example'
access_token = 'your_oauth2_access_token'

response = add_coursework_attachment(course_id, coursework_id, file_id, access_token)
if response:
    print("Attachment added successfully:", response)
```

- The function `add_coursework_attachment` takes in `course_id`, `coursework_id`, `file_id`, and `access_token` as parameters.
- It constructs the URL for the API endpoint and sets up the headers with the authorization token.
- The payload includes the `fileId`, a title, and details of the Drive file to attach.
- The `requests.post` method is used to send the POST request to the API.
- Error handling is implemented using `try-except` blocks to catch HTTP errors and other exceptions.

Google API Documentation

```
- The function returns the JSON response from the API if the request is successful.
```

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

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

Common Use Cases
1. **Remove Unnecessary Files**: Delete attachments that are no longer needed.
2. **Clean Up Course Materials**: Clear out attachments that are outdated or have been superseded by newer versions.
3. **Course Reset**: Remove all attachments to prepare a course for a new term by starting with a clean slate.

Example Request
```http
DELETE
https://api.example.com/classroom.courses/courseId.courses.courseWork/courseWorkId.addOnAttachments/attachmentId
```

Common Parameters
- **courseId** (string): The unique identifier for the course.
- **courseWorkId** (string): The unique identifier for the course work item.
- **attachmentId** (string): The unique identifier for the attachment to be deleted.

Example Request with Placeholder Values
```http
DELETE https://api.example.com/classroom.courses/12345.courses.courseWork/67890.addOnAttachments/112233
```

Example Code

```python
import requests

def delete_course_work_attachment(course_id, course_work_id, attachment_id, access_token):
    """
    Deletes an attachment from a course work.

    :param course_id: The ID of the course.
    :param course_work_id: The ID of the course work.
    :param attachment_id: The ID of the attachment to delete.
    :param access_token: The access token for authentication.
```

```
    :return: Response from the API.
    """
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/addOnAttachments/{attachm
ent_id}"

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

    try:
        response = requests.delete(url, headers=headers)
        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 Exception as err:
        print(f"An error occurred: {err}")

# Example usage
course_id = "1234567890"
course_work_id = "9876543210"
attachment_id = "0987654321"
access_token = "your_access_token_here"

response = delete_course_work_attachment(course_id, course_work_id, attachment_id, access_token)
if response:
    print("Attachment deleted successfully:", response)
```

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

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

Common Use Cases:
- Retrieve a list of attachments for a specific assignment in a course.
- Integrate with learning management systems to display assignments with their associated files.
- Verify the presence and status of attachments for coursework.

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

- **courseId**: `1234567890`
- **courseWorkId**: `0987654321`

Common Parameters:
- **courseId** (string): The ID of the course.
- **courseWorkId** (string): The ID of the coursework (assignment).
- **pageSize** (integer, optional): The maximum number of attachments to return.
- **pageToken** (string, optional): The token for the next page of results.

Example Code

```python
import requests
import json

# Define the API endpoint and the base URL
base_url = "https://classroom.googleapis.com/v1"
endpoint =
"/courses/{course_id}/courseWork/{coursework_id}/studentSubmissions/{student_submission_id}/addOnAttachments"

# Define the parameters for the request
course_id = "course12345"  # Replace with a valid course ID
coursework_id = "coursework67890"  # Replace with a valid coursework ID
student_submission_id = "submission111222333"  # Replace with a valid submission ID
```

```
# Define the API key or access token for authentication
api_key = "your_api_key_here"  # Replace with your actual API key or access token

# Construct the full URL for the request
url = base_url + endpoint.format(
    course_id=course_id,
    coursework_id=coursework_id,
    student_submission_id=student_submission_id
)

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

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
    data = response.json()

    # Print the response data
    print(json.dumps(data, indent=4))

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 other request errors
    print(f"Request error occurred: {req_err}")
except json.JSONDecodeError as json_err:
    # Handle JSON decoding errors
    print(f"JSON decoding error occurred: {json_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 of GET classroom.courses.courseWork.addOnAttachments.studentSubmissions.get

Common Use Cases:
- Retrieve a list of student submissions for a specific assignment.
- Check the status of student attachments for a given coursework.
- Verify if a student has submitted the required attachments for an assignment.

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

Common Parameters:
- **courseId** (Path Parameter): The ID of the course.
- **courseWorkId** (Path Parameter): The ID of the coursework.
- **accessToken** (Header Parameter): The OAuth 2.0 token for authorization.

Example Code

```python
import requests

def get_student_submissions_with_attachments(course_id, course_work_id, student_id):
    """
    Fetch student submissions with attachments for a specific course work.
    :param course_id: ID of the course.
    :param course_work_id: ID of the course work.
    :param student_id: ID of the student.
    :return: Response content if successful, otherwise an error message.
    """
                                                                                url                    =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/studentSubmissions/{student_id}/attachments"
```

```
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN'  # Replace with your actual access token
    }

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses
        return response.json()  # Return JSON content if successful
    except requests.exceptions.HTTPError as http_err:
        return f"HTTP error occurred: {http_err}"
    except requests.exceptions.RequestException as req_err:
        return f"Error occurred: {req_err}"
    except Exception as err:
        return f"An unexpected error occurred: {err}"

# Example usage
course_id = '1234567890'
course_work_id = '9876543210'
student_id = '123456789'

result = get_student_submissions_with_attachments(course_id, course_work_id, student_id)
print(result)
```

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:
- Update the submission status of assignments.
- Modify metadata of submitted assignments.
- Correct errors in previously submitted work.

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

{
  "state": "PENDING",
  "attachments": [
    {
      "driveFile": {
        "driveFileId": "123456789abcdefghijklmnopqrstuvwxyz",
        "title": "Updated Assignment"
      }
    }
  ]
}
```

Common Parameters:
- **courseId** (path parameter): The ID of the course.
- **courseWorkId** (path parameter): The ID of the coursework.
- **addOnAttachmentId** (path parameter): The ID of the add-on attachment.
- **studentId** (path parameter): The ID of the student.
- **state** (optional): The new state of the submission (e.g., "PENDING", "TURNED_IN").
- **attachments** (optional): An array of attachments to update or add to the submission.

Example Code

```python
import requests

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

# Define the parameters (realistic values)
courseId = "1234567890"
courseWorkId = "abcdefghij"
studentId = "student123"
attachmentId = "attachment456"
newSubmission = {
    "status": "TURNPLDED"
}

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

# Define the patch request payload
payload = {
    "onAttachments": {
        "attachments": [
            {
                "id": attachmentId,
                "description": "New attachment description"
            }
        ]
    }
}

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

    # Check for successful response
    if response.status_code == 200:
        print("Submission updated successfully.")
        print(response.json())
    else:
        print(f"Failed to update submission. Status code: {response.status_code}")
        print(response.json())

except requests.exceptions.RequestException as e:
    # Handle any errors that occur during the request
```

```
    print(f"An error occurred: {e}")
```

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

Technical Description for GET classroom.courses.students.get

Common Use Cases:
1. Fetching a list of students enrolled in a specific course.
2. Retrieving student details for administrative purposes.
3. Integrating with student management systems to synchronize data.

Example Request:
```http
GET https://api.example.com/classroom/courses/{course_id}/students
```

Common Parameters:
1. **course_id (required)**: The unique identifier for the course.
2. **limit (optional)**: The maximum number of student records to return. Defaults to 100.
3. **offset (optional)**: The number of records to skip before starting to collect the result set. Defaults to 0.
4. **filter (optional)**: A query string to filter students based on specific criteria (e.g., active, inactive).

Example Code

```python
import requests

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

    # Set the 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 response status code is successful
```

```
        if response.status_code == 200:
            # Parse the JSON response
            students = response.json()
            return students
        else:
            # Handle non-successful response status codes
            response.raise_for_status()

    except requests.exceptions.RequestException as e:
        # Handle any errors that occur during the request
        print(f"An error occurred: {e}")
        return None

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

students = get_students(course_id, access_token)
if students:
    print(students)
```

```python
import requests

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

    # Set the 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 response status code is successful
        if response.status_code == 200:
            # Parse the JSON response
            students = response.json()
            return students
        else:
            # Handle non-successful response status codes
            response.raise_for_status()

    except requests.exceptions.RequestException as e:
        # Handle any errors that occur during the request
```

```
        print(f"An error occurred: {e}")
        return None

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

students = get_students(course_id, access_token)
if students:
    print(students)
```

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

Technical Description for GET classroom.courses.students.list

Common Use Cases
- Retrieve a list of students enrolled in a specific course.
- Monitor student enrollment status and demographics.
- Generate reports on student participation and performance.

Example Request
```
GET /classroom/courses/{course_id}/students.list
```

Common Parameters
- **course_id**: (Required) The unique identifier for the course.
- **limit**: (Optional) The maximum number of students to return in the list.
- **offset**: (Optional) The number of students to skip before starting to collect the result set.
- **sort_by**: (Optional) The field to sort the students by (e.g., name, enrollment_date).
- **filter**: (Optional) Filters to apply to the student list (e.g., active, inactive).

Example Code

```python
import requests

def list_students_in_course(course_id, page_token=None):
    """
    Lists students in a specific course using the classroom.courses.students.list API.

    Args:
    course_id (str): The ID of the course.
    page_token (str, optional): The token to retrieve the next page of results.

    Returns:
    dict: The response from the API containing the list of students.
    """
    url = "https://classroom.googleapis.com/v1/courses/{}/students".format(course_id)
    params = {
        'pageToken': page_token
```

```
    }

    try:
        response = requests.get(url, params=params)
        response.raise_for_status()  # Raise an exception for HTTP errors
        return response.json()  # Return the JSON response
    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}")

# Example usage:
course_id = "1234567890abcdef"
page_token = None
students = list_students_in_course(course_id, page_token)
if students:
    print(students)
```

This code defines a function `list_students_in_course` that takes a `course_id` and an optional `page_token` to retrieve the list of students in a specific course using the `classroom.courses.students.list` API. The function includes proper error handling for various exceptions that might be raised during the API call and returns the JSON response if successful.

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

Technical Description for DELETE classroom.courses.students.delete

Common Use Cases
- Removing a student from a specific course.
- Updating course enrollment records.
- Handling student withdrawals or transfers.

Example Request
```http
DELETE /classroom/courses/{courseId}/students/{studentId}
Authorization: Bearer YOUR_ACCESS_TOKEN
Content-Type: application/json
```

Common Parameters
- **courseId** (Path Parameter, Required): The unique identifier for the course from which the student will be removed.
- **studentId** (Path Parameter, Required): The unique identifier for the student to be removed from the course.
- **Authorization** (Header, Required): Bearer token for authentication.
- **Content-Type** (Header, Optional): Typically `application/json`, specifies the format of the request body.

Example Code

```python
import requests

def delete_student_from_course(course_id, student_id):
    """
    Deletes a student from a course using the classroom.courses.students.delete API endpoint.

    Args:
    course_id (str): The ID of the course.
    student_id (str): The ID of the student to be deleted.

    Returns:
    dict: The response from the API.
    """
    # Define the URL for the API endpoint
```

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

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

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

        # Check if the request was successful
        if response.status_code == 200:
            return response.json()
        else:
            # Handle different error codes
            if response.status_code == 401:
                return {"error": "Unauthorized: Invalid or missing access token"}
            elif response.status_code == 403:
                return {"error": "Forbidden: Insufficient permissions"}
            elif response.status_code == 404:
                return {"error": "Not Found: Course or student ID invalid"}
            else:
                return {"error": f"Unexpected error: {response.status_code}"}

    except requests.exceptions.RequestException as e:
        # Handle network-related errors
        return {"error": f"Network error: {str(e)}"}

# Example usage
course_id = "course123"
student_id = "student456"
response = delete_student_from_course(course_id, student_id)
print(response)
```

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

POST classroom.courses.students.create

Common Use Cases:
- Enrolling a student in a specific course.
- Adding a new student to a pre-existing course roster.
- Automating the enrollment process via an integration with a student information system.

Example Request:
```json
POST /classroom/courses/students/create
Content-Type: application/json

{
  "course_id": "CSC101",
  "student_id": "STU2023123",
  "enrollment_date": "2023-10-01"
}
```

Common Parameters:
- **course_id** (string): The unique identifier for the course in which the student is to be enrolled.
- **student_id** (string): The unique identifier for the student to be added.
- **enrollment_date** (string, format: yyyy-MM-dd): The date on which the student is being enrolled.

Example Code

```python
import requests

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

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

# Define the payload with realistic parameter names and values
payload = {
    "userId": "user@example.com",
    "courseId": "123456789",
    "role": "STUDENT",
    "state": "ACTIVE"
}

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

# Check for HTTP errors
if response.status_code == 200:
    print("Student created successfully")
    print(response.json())  # Print the response JSON
elif response.status_code == 400:
    print("Bad request: Check your parameters and try again.")
    print(response.json())  # Print the error details
elif response.status_code == 401:
    print("Unauthorized: Invalid or missing access token.")
elif response.status_code == 403:
    print("Forbidden: Insufficient permissions to perform the action.")
elif response.status_code == 404:
    print("Not found: The course or user may not exist.")
else:
    print(f"An unexpected error occurred: {response.status_code}")
    print(response.json())  # Print the error details
```

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

API Endpoint: POST classroom.courses.announcements.modifyAssignees

Use Cases:
- Modify the list of assignees for a specific course announcement.
- Update the assignee details when changes are needed.
- Ensure that the correct individuals are notified about announcements.

Example Request:
```json
POST /classroom/courses/announcements/modifyAssignees
Content-Type: application/json

{
  "courseId": "12345",
  "announcementId": "67890",
  "assignees": ["student1@example.com", "student2@example.com"]
}
```

Common Parameters:
- **courseId** (string): The unique identifier for the course.
- **announcementId** (string): The unique identifier for the announcement within the course.
- **assignees** (array of strings): The list of email addresses of the students to whom the announcement is assigned.

Example Code

```python
import requests

def modify_announcement_assignees(course_id, announcement_id, users_to_add=None, users_to_remove=None):
    """
    Modify the assignees of an announcement in a course.

    :param course_id: ID of the course.
    :param announcement_id: ID of the announcement.
```

```
    :param users_to_add: List of user IDs to add as assignees.
    :param users_to_remove: List of user IDs to remove as assignees.
    """
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/modifyAssignees"

    # Prepare the request payload
    payload = {
        'addAssignees': users_to_add,
        'removeAssignees': users_to_remove
    }

    # Set up headers with necessary authorization
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN'
    }

    try:
        # Make the API request
        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 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
course_id = '1234567890'
announcement_id = '9876543210'
users_to_add = ['user1@example.com', 'user2@example.com']
users_to_remove = ['user3@example.com']

response = modify_announcement_assignees(course_id, announcement_id, users_to_add, users_to_remove)
if response:
    print("Assignees modified successfully:", response)
```

```python
# Explanation:
# - The function `modify_announcement_assignees` takes in the course ID, announcement ID, and optional lists of
users to add or remove.
```

```
# - The URL for the API endpoint is constructed using the course ID and announcement ID.
# - The payload includes the lists of users to add and remove as assignees.
# - Headers are set up with the necessary authorization token.
# - The `requests.post` method is used to send the API request.
# - Error handling is implemented to catch and print HTTP errors, request exceptions, and other unexpected errors.
# - The example usage demonstrates how to call the function with realistic parameter values.
```

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

Technical Description

Endpoint: `PATCH classroom.courses.announcements.patch`

Common Use Cases:
1. Update the details of an existing announcement in a course.
2. Modify the content or status of an announcement without deleting it.
3. Change the scheduled release date or time of an announcement.

Example Request:
```
PATCH /classroom/courses/{courseId}/announcements/{announcementId}
Content-Type: application/json

{
  "title": "Updated Title",
  "content": "This is the updated content of the announcement.",
  "release_date": "2023-12-31T23:59:59Z"
}
```

Common Parameters:
1. `courseId` (Path Parameter): The unique identifier for the course.
2. `announcementId` (Path Parameter): The unique identifier for the announcement to be updated.
3. `title` (Body Parameter, Optional): The updated title of the announcement.
4. `content` (Body Parameter, Optional): The updated content of the announcement.
5. `release_date` (Body Parameter, Optional): The updated scheduled release date and time for the announcement in ISO 8601 format.
6. `status` (Body Parameter, Optional): The updated status of the announcement (e.g., 'published', 'draft').

Example Code

```python
import requests

# Define the API endpoint and headers
```

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

# Define the data to be patched
data = {
    "courseId": "1234567890",
    "announcementId": "9876543210",
    "updateMask": "text",
    "resource": {
        "text": "This is the updated announcement text."
    }
}

# Make the PATCH request
try:
    response = requests.patch(url, headers=headers, json=data)
    response.raise_for_status()  # Raise an exception for HTTP errors
    print("Announcement updated successfully:", response.json())
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as err:
    print(f"An error occurred: {err}")
```

Explanation:
1. **Importing Requests Library**: We import the `requests` library to handle HTTP requests.
2. **Defining API Endpoint and Headers**: We set the URL for the API endpoint and include headers for authorization using a Bearer token and content type.
3. **Defining Data to be Patched**: We specify the course ID, announcement ID, update mask, and the new text for the announcement.
4. **Making the PATCH Request**: We use the `requests.patch` method to send the PATCH request. We include error handling to catch HTTP errors and other request exceptions. If the request is successful, we print the response; otherwise, we print the error message.

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

Technical Description for DELETE classroom.courses.announcements.delete

Common Use Cases:
- Deleting an announcement from a specific course within a classroom.
- Removing outdated or incorrect announcements.
- Cleaning up announcements no longer relevant to the course.

Example Request:
```http
DELETE /classroom/courses/123/announcements/456
```
- Replace `/123` with the course ID.
- Replace `/456` with the announcement ID.

Common Parameters:
- `courseId` (Path Parameter): The unique identifier for the course.
- `announcementId` (Path Parameter): The unique identifier for the announcement to be deleted.
- `Authorization` (Header): Bearer token for authentication.

Example Code

```python
import requests
import json

def delete_course_announcement(course_id, announcement_id, access_token):
    """
    Deletes an announcement from a specific course.

    Params:
    - course_id (str): The ID of the course.
    - announcement_id (str): The ID of the announcement to delete.
    - access_token (str): The access token for authentication.
```

```
    Returns:
    - dict: Response from the API.
    """
    # Define the API endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}"

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

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

    # Check for HTTP errors
    if response.status_code == 204:
        print("Announcement deleted successfully.")
        return {"status": "success", "message": "Announcement deleted successfully."}

    elif response.status_code == 401:
        print("Unauthorized: Invalid or expired access token.")
        return {"status": "error", "message": "Unauthorized: Invalid or expired access token."}

    elif response.status_code == 404:
        print(f"Announcement with ID {announcement_id} not found.")
        return {"status": "error", "message": "Announcement not found."}

    else:
        print(f"Error: {response.status_code}, {response.text}")
        return {"status": "error", "message": f"Error: {response.status_code}, {response.text}"}

# Example usage
if __name__ == "__main__":
    course_id = "1234567890"
    announcement_id = "0987654321"
    access_token = "your_access_token_here"

    result = delete_course_announcement(course_id, announcement_id, access_token)
    print(result)
```

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.
- Fetch details of individual announcements for monitoring and updates.
- Integrate with educational platforms to display announcements on student dashboards.

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

Common Parameters:
- `courseId` (required): The unique identifier for the course whose announcements are to be retrieved.
- `filter` (optional): A string to filter announcements by specific criteria, such as date or author.
- `limit` (optional): The maximum number of announcements to return in a single response, useful for pagination.
- `offset` (optional): The number of announcements to skip before beginning to collect the result, useful for pagination.

Example Code

```python
import requests

def get_course_announcements(course_id, access_token):
    # API endpoint URL
    url = 'https://classroom.googleapis.com/v1/courses/{}/announcements'.format(course_id)

    # Headers for the request
    headers = {
        'Authorization': 'Bearer ' + access_token
    }

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

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

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

        # Return the announcements data
        return data

    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'  # Replace with a valid course ID
access_token = 'YOUR_ACCESS_TOKEN'  # Replace with a valid access token
announcements = get_course_announcements(course_id, access_token)
print(announcements)
```

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:
1. Retrieve a list of announcements for a specific course.
2. Allow students to stay updated on course-related news.
3. Enable teachers to review and manage course announcements.

Example Request:
```http
GET /classroom/courses/<course_id>/announcements.list
```

Common Parameters:
- `course_id` (String): The unique identifier for the course.
- `pageToken` (String, optional): A token to specify a page of results to retrieve.
- `maxResults` (Integer, optional): The maximum number of results to return per page.
- `orderBy` (String, optional): Specifies the ordering of announcements (e.g., "updateTime desc").

Example Code

```python
import requests

def get_announcements(course_id, page_token=None):
    """
    Fetches announcements for a given course ID.

    Parameters:
    - course_id (str): The ID of the course.
    - page_token (str, optional): The token for pagination.

    Returns:
    - dict: The list of announcements.
    """
    url = "https://classroom.googleapis.com/v1/courses/{}/announcements".format(course_id)
```

```
    params = {
        'pageToken': page_token
    }
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN'  # Replace with actual access token
    }

    try:
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()  # Raise 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 err:
        print(f"Error occurred: {err}")
    except ValueError as json_err:
        print(f"JSON decode error: {json_err}")

    return None  # Return None if there is an error

# Example usage
course_id = '1234567890'
announcements = get_announcements(course_id)
if announcements:
    print(announcements)
else:
    print("Failed to fetch announcements.")
```

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:
1. **Instructors**: To notify students about important updates, deadlines, or changes in course schedules.
2. **Administrators**: To disseminate urgent information or administrative announcements to specific courses.
3. **Teaching Assistants**: To relay information from instructors to students when the instructor is unavailable.

Example Request:
```http
POST /classroom.courses.announcements.create
Content-Type: application/json

{
  "courseId": "12345",
  "title": "Assignment Deadline Extended",
  "content": "The deadline for the midterm assignment has been extended to December 15, 2023.",
  "priority": "high",
  "sendingUserId": "67890"
}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course to which the announcement is directed.
- `title` (string): The title of the announcement.
- `content` (string): The main body of the announcement.
- `priority` (string): The priority level of the announcement (e.g., "low", "medium", "high").
- `sendingUserId` (string): The unique identifier of the user sending the announcement.

Example Code

```python
import requests

# Define the API endpoint and headers
url = 'https://classroom.googleapis.com/v1/courses/{courseId}/announcements'
headers = {
```

```
    'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
    'Content-Type': 'application/json'
}

# Define the parameters for the announcement
data = {
    "announcement": {
        "text": "Welcome to the new semester! Please read the syllabus carefully.",
        "state": "PUBLISHED"
    }
}

# Replace 'YOUR_ACCESS_TOKEN' and 'YOUR_COURSE_ID' with actual values
course_id = 'YOUR_COURSE_ID'
url = url.format(courseId=course_id)

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

    # Check for successful response
    if response.status_code == 200:
        print("Announcement created successfully!")
        print(response.json())
    else:
        # Handle different error cases
        if response.status_code == 400:
            print("Bad Request: Check the parameters.")
        elif response.status_code == 401:
            print("Unauthorized: Check your access token.")
        elif response.status_code == 403:
            print("Forbidden: You do not have permission to create announcements.")
        elif response.status_code == 404:
            print("Not Found: The course ID is invalid.")
        else:
            print(f"An error occurred: {response.status_code}")
            print(response.json())

except requests.exceptions.RequestException as e:
    # Handle network-related errors
    print(f"An error occurred: {e}")
```

- This code sets up a POST request to the `classroom.courses.announcements.create` endpoint.
- It uses realistic parameter names and values, such as `text` for the announcement content and `state` for the publication status.
- Error handling is included to manage different HTTP status codes and network-related exceptions.

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

GET classroom.courses.announcements.getAddOnContext

Common Use Cases:
- Fetch contextual information for displaying announcements within an educational app.
- Retrieve data to integrate with third-party tools for classroom announcements.
- Support for school administrators to manage and customize announcement notifications.

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

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `announcementId` (string): The unique identifier for the announcement within the course.
- `addOnContext` (optional, string): Additional context for the add-on, useful for customizing the response.
- `fields` (optional, string): Selector specifying which fields to include in a partial response.

Example Code

```python
import requests

def get_add_on_context(course_id, announcement_id):
    # Define the API endpoint
       url  =  "https://classroom.googleapis.com/v1/courses/{}/announcements/{}/addOnContext".format(course_id, announcement_id)

    # Define the headers with authentication token
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"
    }
```

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

        # Raise an exception if the request returned an unsuccessful status code
        response.raise_for_status()

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

        # Return the parsed 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
course_id = "1234567890"
announcement_id = "9876543210"
add_on_context = get_add_on_context(course_id, announcement_id)
if add_on_context:
    print(add_on_context)
```

This Python script makes a GET request to the `classroom.courses.announcements.getAddOnContext` API endpoint. It includes error handling for HTTP errors and other request exceptions. Replace `"YOUR_ACCESS_TOKEN"` with a valid access token for authenticating the request. The function `get_add_on_context` takes `course_id` and `announcement_id` as parameters and returns the parsed JSON response if successful.

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

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

Common Use Cases:
1. Retrieve a list of attachments added to course announcements.
2. Review specific attachments for auditing or compliance purposes.
3. Retrieve metadata about attachments for further processing or filtering.

Example Request:
```
GET
https://classroom.googleapis.com/classroom/v1/courses/COURSE_ID/announcements/ANNOUNCEMENT_ID/addOnAttachments
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
1. **courseId** (required): The identifier for the course. Example: `COURSE_ID`
2. **announcementId** (required): The identifier for the announcement. Example: `ANNOUNCEMENT_ID`
3. **pageToken** (optional): The token for the next page of results. Example: `TOKEN_VALUE`
4. **pageSize** (optional): Maximum number of attachments to return. Example: `10`

Example Code

```python
import requests

def get_announcement_attachments(course_id, announcement_id):
    """
    Fetch the list of attachments for a specific announcement in a course.

    Args:
    course_id (str): The ID of the course.
    announcement_id (str): The ID of the announcement.
```

```
    Returns:
    list: A list of attachment details.
    """
    # Define the API endpoint URL
                                                                                    url                     =
f'https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments'

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

    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
        attachments = response.json().get('attachments', [])

        return attachments

    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}')

    return []

# Example usage
course_id = '1234567890'
announcement_id = '9876543210'
attachments = get_announcement_attachments(course_id, announcement_id)
print(attachments)
```

In this example:
- The `get_announcement_attachments` function takes `course_id` and `announcement_id` as parameters.
- It constructs the API endpoint URL using the provided IDs.
- The `headers` dictionary includes an authorization header with a placeholder for the actual access token.
- The `requests.get` method is used to make the GET request to the API.
- Error handling is included to catch and print HTTP errors and other request exceptions.
- The function returns a list of attachments if the request is successful, or an empty list if an error occurs.

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

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

Common Use Cases
- Retrieve a list of attachments added to a specific announcement in a course.
- Verify the status and details of attachments for a particular announcement.
- Assist in troubleshooting issues related to announcement attachments.

Example Request

```http
GET https://classroom.googleapis.com/v1/courses/{courseId}/announcements/{announcementId}/attachments
Authorization: Bearer {ACCESS_TOKEN}
```

Common Parameters

- **courseId** (string, required): The unique identifier for the course.
- **announcementId** (string, required): The unique identifier for the announcement within the course.
- **access_token** (string, required): The OAuth 2.0 access token that authorizes the request.

Example Code

```python
import requests

# Define the API endpoint
endpoint = "https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments"

# Define the parameters for the request
course_id = "1234567890"  # Replace with a valid course ID
announcement_id = "0987654321"  # Replace with a valid announcement ID

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

```
# Make the API request
response = requests.get(endpoint.format(course_id=course_id, announcement_id=announcement_id), headers=headers)

# Check for HTTP errors
if response.status_code != 200:
    # Handle different types of errors
    if response.status_code == 401:
        print("Unauthorized: Check your access token.")
    elif response.status_code == 403:
        print("Forbidden: You may not have access to this resource.")
    elif response.status_code == 404:
        print("Not Found: The course or announcement may not exist.")
    else:
        print(f"Error: {response.status_code}")
    # Exit the program or handle the error accordingly
    exit(1)

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

# Print the attachments (for demonstration purposes)
print(data)
```

This code makes a GET request to the `classroom.courses.announcements.addOnAttachments.get` endpoint, using realistic parameter names and values. It includes proper error handling for different HTTP status codes and prints the attachments if the request is successful.

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

POST classroom.courses.announcements.addOnAttachments.create

Common Use Cases:
- Adding attachments to announcements for course-related documents.
- Distributing supplementary materials like PDFs, images, or videos.
- Updating announcements with new attachments for ongoing course information.

Example Request:
```http
POST /classroom/courses/announcements/addOnAttachments/create
Content-Type: application/json

{
  "courseId": "course_12345",
  "announcementId": "announcement_67890",
  "attachment": {
    "fileName": "lecture_notes.pdf",
    "fileContent": "base64_encoded_content"
  }
}
```

Common Parameters:
- `courseId` (string): The unique identifier for the course.
- `announcementId` (string): The unique identifier for the announcement.
- `attachment` (object):
  - `fileName` (string): The name of the file to be attached.
  - `fileContent` (string): The content of the file, typically base64 encoded.

Example Code

```python
import requests

# Define the API endpoint and the required headers
url = "https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments"
headers = {
```

```
    "Authorization": "Bearer YOUR_ACCESS_TOKEN",
    "Content-Type": "application/json"
}

# Define the parameters for the request
course_id = "1234567890"
announcement_id = "0987654321"
file_id = "file_id_example"
mime_type = "application/pdf"
title = "Example PDF Attachment"

# Define the payload for the request
payload = {
    "title": title,
    "fileId": file_id,
    "mimeType": mime_type
}

# Make the POST request to create an attachment
try:
    response = requests.post(url.format(course_id=course_id, announcement_id=announcement_id), headers=headers,
json=payload)
    response.raise_for_status()  # Raise an exception for HTTP errors
    print("Attachment created successfully:", 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.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 Endpoint: PATCH classroom.courses.announcements.addOnAttachments.patch

Common Use Cases:
- Adding new attachments to an existing announcement.
- Updating existing attachments in an announcement.
- Removing specific attachments from an announcement.

Example Request:
```http
PATCH /classroom/courses/{courseId}/announcements/{announcementId}/addOnAttachments
Content-Type: application/json

{
  "attachments": [
    {
      "fileId": "12345",
      "fileName": "new_file.txt"
    }
  ]
}
```

Common Parameters:
- **courseId (Path Parameter)**: The unique identifier for the course.
- **announcementId (Path Parameter)**: The unique identifier for the announcement within the course.
- **attachments (Request Body)**:
  - **fileId (String)**: The unique identifier of the file to be attached.
  - **fileName (String)**: The name of the file to be attached.

Example Code

```python
import requests

def patch_course_announcements(announcement_id, attachment_id, access_token):
    """
    Patches an announcement in Google Classroom to add an attachment.
```

```
    Parameters:
    announcement_id (str): The ID of the announcement to be updated.
    attachment_id (str): The ID of the attachment to be added.
    access_token (str): The OAuth 2.0 access token for authorization.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{announcement_id}/announcements/{attachment_id}"

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

    # Define the payload for the patch request
    payload = {
        'addOnAttachments': [
            {
                'fileId': attachment_id
            }
        ]
    }

    try:
        response = requests.patch(url, headers=headers, json=payload)
        response.raise_for_status()  # Raises an HTTPError for bad responses (4xx and 5xx)
        print("Announcement updated successfully")
        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
announcement_id = '1234567890'
attachment_id = '0987654321'
access_token = 'YOUR_ACCESS_TOKEN_HERE'

patch_course_announcements(announcement_id, attachment_id, access_token)
```

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 DELETE classroom.courses.announcements.addOnAttachments.delete

Common Use Cases:
1. Remove a specific attachment from a course announcement.
2. Clean up outdated or erroneous attachments to maintain data integrity.
3. Facilitate the removal of attachments that are no longer relevant to the content of the announcement.

Example Request:
```
DELETE /v1/classroom/courses/{courseId}/announcements/{announcementId}/addOnAttachments/{attachmentId}
```

Common Parameters:
- **courseId**: The unique identifier for the course.
- **announcementId**: The unique identifier for the announcement.
- **attachmentId**: The unique identifier for the attachment to be deleted.

Example Code

```python
import requests

# Function to delete attachments from a classroom announcement
def delete_attachments_from_announcement(announcement_id, file_id, access_token):
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/announcements/{announcement_id}/attachments/{file_id}
"
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'application/json'
    }
    response = requests.delete(url, headers=headers)

    # Handle response and errors
    if response.status_code == 204:
        print("Attachment deleted successfully.")
    elif response.status_code == 401:
```

```
        print("Unauthorized: Check your access token.")
    elif response.status_code == 403:
        print("Forbidden: You do not have permission to delete this attachment.")
    elif response.status_code == 404:
        print("Not Found: The specified announcement or attachment does not exist.")
    else:
        print(f"Error: {response.status_code} - {response.text}")

# Example usage
course_id = '1234567890'
announcement_id = '9876543210'
file_id = 'file_123'
access_token = 'your_access_token_here'

delete_attachments_from_announcement(announcement_id, file_id, access_token)
```

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

Endpoint: PATCH classroom.courses.topics.patch

Common Use Cases:
- Update the title or description of a specific topic within a course.
- Modify the order of topics within a course.
- Add or remove resources from a topic.

Example Request:
```
PATCH /classroom/courses/123/topics/456 HTTP/1.1
Host: api.example.com
Content-Type: application/json

{
  "title": "Updated Topic Title",
  "description": "Updated description of the topic."
}
```

Common Parameters:
- **course_id** (integer): The unique identifier for the course.
- **topic_id** (integer): The unique identifier for the topic within the course.
- **title** (string, optional): The updated title of the topic.
- **description** (string, optional): The updated description of the topic.
- **order** (integer, optional): The new order position of the topic within the course.
- **resources** (array of strings, optional): List of resources to be added or removed from the topic.

Example Code

```python
import requests

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

# Define realistic parameter names and values
```

```
course_id = "1234567890"
topic_id = "12345"
update_body = {
    "topic": "New Topic Name",
    "description": "Updated description for the topic."
}

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

# Make the PATCH request
response = requests.patch(f"{url}", json=update_body, headers=headers)

# Check for HTTP errors
if response.status_code == 200:
    print("Topic updated successfully.")
    print("Response:", response.json())
elif response.status_code == 400:
    print("Bad Request: Check your request parameters.")
    print("Error Response:", response.json())
elif response.status_code == 401:
    print("Unauthorized: Check your access token.")
    print("Error Response:", response.json())
elif response.status_code == 403:
    print("Forbidden: You do not have permission to update this topic.")
    print("Error Response:", response.json())
elif response.status_code == 404:
    print("Not Found: The course or topic does not exist.")
    print("Error Response:", response.json())
else:
    print("An unexpected error occurred.")
    print("Error Response:", response.json())
```

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

Technical Description for GET classroom.courses.topics.list

Common Use Cases:
1. Fetching a list of topics for a specific course.
2. Retrieving topic details for administrative or educational purposes.
3. Integrating with learning management systems.

Example Request:
```
GET /classroom/courses/topics/list?courseId=12345
Host: api.example.com
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **courseId** (required): The unique identifier for the course.
- **page** (optional): The page number to retrieve (default is 1).
- **limit** (optional): The number of topics to return per page (default is 10).
- **sortBy** (optional): The field to sort the topics by (e.g., name, createdAt).

Example Code

```python
import requests

def list_topics(course_id, api_key):
    """
    Fetches the list of topics for a given course.

    Args:
    course_id (str): The ID of the course.
    api_key (str): The API key for authentication.

    Returns:
    dict: The response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/topics"
```

```
    headers = {
        'Authorization': f'Bearer {api_key}',
        '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 Exception as err:
        print(f"Other error occurred: {err}")
    return None

# Example usage
course_id = "1234567890"
api_key = "your_api_key_here"
topics = list_topics(course_id, api_key)
if topics:
    print("Topics:", topics)
```

This code defines a function `list_topics` that takes a `course_id` and an `api_key` as parameters. It constructs the URL for the API endpoint, sets the necessary headers for authorization and accepts JSON format, and makes a GET request to the API. The function includes error handling for HTTP errors and other potential exceptions, printing an error message if any occur. If the request is successful, it returns the JSON response; otherwise, it returns `None`.

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

Technical Description for GET classroom.courses.topics.get

Common Use Cases
- Fetching topics for a specific course.
- Retrieving a list of topics to display in a course syllabus.
- Integrating course topics with learning management systems.

Example Request
```
GET /classroom/courses/{courseId}/topics
```

Common Parameters
- **courseId**: The unique identifier for the course whose topics are being requested.
- **page**: The page number to retrieve (optional, used for pagination).
- **limit**: The maximum number of topics to return per page (optional, used for pagination).
- **filter**: A query string to filter topics based on specific criteria (optional).
- **sort**: A query string to sort topics by a specific field (optional).

Example request with placeholder values:
```
GET /classroom/courses/12345/topics?page=1&limit=10&filter=active&sort=createdAt
```

Example Code

```python
import requests

def get_course_topics(course_id, access_token):
    # Define the API endpoint
    url = "https://classroom.googleapis.com/v1/courses/{}/topics".format(course_id)

    # Set up the headers with the access token for authorization
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Accept': 'application/json'
```

```
    }

    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
        topics = response.json()

        # 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
course_id = "1234567890"
access_token = "your_access_token_here"
topics = get_course_topics(course_id, access_token)
print(topics)
```

This code defines a function `get_course_topics` that takes a `course_id` and an `access_token` as parameters. It constructs the API request URL, sets up the necessary headers, and makes a GET request to the API. The function includes error handling for HTTP errors and other exceptions, printing error messages if they occur. Finally, it returns the list of topics if the request is successful.

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

Technical Description for DELETE classroom.courses.topics.delete

Common Use Cases
- Remove a specific topic from a course.
- Delete a topic that is no longer relevant to the course curriculum.
- Admin or instructor actions to manage course content dynamically.

Example Request
```http
DELETE /classroom/courses/123/topics/456
Authorization: Bearer <access_token>
```

Common Parameters
- **course_id** (Path Parameter, Required): The unique identifier for the course from which the topic will be deleted.
- **topic_id** (Path Parameter, Required): The unique identifier for the topic to be deleted.
- **Authorization** (Header, Required): Bearer token for authenticating the request.

Example Code

```python
import requests

def delete_topic(course_id, topic_id, api_key):
    """
    Deletes a topic from a course using the classroom.courses.topics.delete API endpoint.

    :param course_id: The ID of the course.
    :param topic_id: The ID of the topic to delete.
    :param api_key: The API key for authentication.
    :return: Response from the API.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/topics/{topic_id}"
    headers = {
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }
```

```
    try:
        response = requests.delete(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}")  # Print HTTP errors
    except requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")  # Print other request exceptions
    except Exception as e:
        print(f"An unexpected error occurred: {e}")  # Print unexpected errors

# Example usage:
# api_key = 'your_api_key_here'
# course_id = 'course_12345'
# topic_id = 'topic_67890'
# delete_topic(course_id, topic_id, api_key)
```

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

Technical Description for POST classroom.courses.topics.create

Common Use Cases:
- Teachers create a new topic for a course to organize course materials.
- Administrators add a new learning module to a specific course.
- Course coordinators structure the course content into manageable topics.

Example Request:
```http
POST /classroom/courses/topics/create
Content-Type: application/json

{
  "course_id": "12345",
  "topic_name": "Introduction to Algorithms",
  "description": "Overview of basic algorithms and data structures."
}
```

Common Parameters:
- **course_id**: (String, Required) Unique identifier for the course to which the topic will be added.
- **topic_name**: (String, Required) The name of the new topic.
- **description**: (String, Optional) A brief description of the topic.

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

Description for POST classroom.courses.teachers.create

Common Use Cases:
- Adding a new teacher to a specific course.
- Assigning multiple teachers to a course for collaborative teaching.

Example Request:
```json
{
  "course_id": "12345",
  "teacher_ids": ["67890", "11223"]
}
```

Common Parameters:
- **course_id**: (String) The unique identifier for the course to which the teachers will be assigned.
- **teacher_ids**: (Array of Strings) A list of unique identifiers for the teachers to be added to the course.

Example Code

```python
import requests

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

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

```
    "teacherId": "teacher_email@example.com"  # The email address of the teacher to add
}

# Function to create a teacher for a course
def create_teacher_for_course(course_id):
    # Include the course ID in the URL
    full_url = url.format(course_id=course_id)

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

        # Check if the response status code indicates success
        if response.status_code == 200:
            print("Teacher successfully added to the course.")
            return response.json()  # Return the response JSON
        else:
            # Handle different types of errors
            if response.status_code == 400:
                print("Bad request: Check the payload for errors.")
            elif response.status_code == 401:
                print("Unauthorized: Check your access token.")
            elif response.status_code == 403:
                print("Forbidden: You do not have permissions to perform this action.")
            elif response.status_code == 404:
                print("Not found: The specified course or teacher does not exist.")
            else:
                print(f"An unexpected error occurred: {response.status_code}")
            return None

    except requests.exceptions.RequestException as e:
        # Handle network-related errors
        print(f"Request failed: {e}")
        return None

# Example usage: add a teacher to a specific course
course_id = "1234567890"  # Replace with an actual course ID
create_teacher_for_course(course_id)
```

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

Endpoint: DELETE classroom.courses.teachers.delete

Common Use Cases:
- Remove a teacher from a specific course.
- Manage teacher assignments to ensure only authorized personnel are listed.
- Update course records by removing outdated or incorrect teacher assignments.

Example Request:
```http
DELETE /classroom/courses/123/teachers/456 HTTP/1.1
Host: api.example.com
Authorization: Bearer your_access_token
```

Common Parameters:
- **courseId** (Path Parameter): The unique identifier for the course from which the teacher is to be removed.
- **teacherId** (Path Parameter): The unique identifier for the teacher to be removed from the course.
- **Authorization** (Header): The access token required to authenticate the request.

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

Common Use Cases:
1. Retrieve a list of teachers associated with a specific course.
2. Verify the teaching staff for a particular course.
3. Integrate course information with other educational applications.

Example Request:
```
GET /api/v1/classroom/courses/teachers
```

Common Parameters:
- `course_id` (required): The unique identifier for the course.
- `limit` (optional): The maximum number of teacher records to return.
- `offset` (optional): The number of records to skip before starting to collect the result set.
- `sort_by` (optional): The field to sort the results by (e.g., 'teacher_name').
- `sort_order` (optional): The order to sort the results in (e.g., 'asc' or 'desc').
- `include_inactive` (optional): A boolean indicating whether to include inactive teachers (default is false).

Example Code

```python
import requests

def get_teachers_for_course(course_id):
    """
    Function to get teachers for a specific course using the classroom.courses.teachers.get API endpoint.
    """
    url = "https://classroom.googleapis.com/v1/courses/{}/teachers".format(course_id)
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"
    }

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses

        # Parse the JSON response
```

```
        teachers = response.json()
        return teachers

    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 ValueError as json_err:
        print(f"JSON decode error: {json_err}")

    return None

# Example usage:
course_id = "1234567890"
teachers = get_teachers_for_course(course_id)
if teachers:
    print("Teachers for the course:", teachers)
else:
    print("Failed to retrieve teachers for the course.")
```

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

Endpoint: GET classroom.courses.teachers.list

Common Use Cases:
- Retrieve a list of teachers assigned to a specific course.
- Verify the teaching staff for a particular class or subject.
- Integrate with other educational systems to ensure accurate teacher assignments.

Example Request:
```
GET /classroom/courses/teachers.list?course_id=12345
```

Common Parameters:
- `course_id` (required): The unique identifier for the course whose teachers are to be listed.
- `limit` (optional): The number of teacher entries to return. Default is 10.
- `offset` (optional): The number of teacher entries to skip. Default is 0.
- `sort` (optional): The field by which to sort the results. Default is by teacher name.

Example Code

```python
import requests

def get_teachers_from_course(course_id, access_token):
    """
    Fetches a list of teachers for a given course using the classroom.courses.teachers.list API endpoint.

    Args:
        course_id (str): The ID of the course.
        access_token (str): The OAuth 2.0 access token.

    Returns:
        list: A list of teacher objects if successful, otherwise an empty list.
    """
    url = 'https://classroom.googleapis.com/v1/courses/{courseId}/teachers'.format(courseId=course_id)
    headers = {
        'Authorization': f'Bearer {access_token}'
    }
```

```
    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)
        data = response.json()
        if 'teachers' in data:
            return data['teachers']
        else:
            print("No teachers found or unexpected response format.")
            return []

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # e.g., 404 Not Found or 500 Internal Server Error
    except requests.exceptions.RequestException as err:
        print(f'Other error occurred: {err}')  # e.g., network issues
    except ValueError as json_err:
        print(f'Error decoding JSON response: {json_err}')

    return []

# Example usage
access_token = 'your_oauth2_access_token_here'
course_id = 'course_12345'
teachers = get_teachers_from_course(course_id, access_token)
print(teachers)
```

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

Endpoint: DELETE classroom.courses.aliases.delete

Common Use Cases:

* Removing an alias from a course to improve URL consistency and user experience.
* Cleaning up deprecated or incorrect aliases that are no longer needed.
* Deleting an alias to enforce a new naming convention for courses.

Example Request:

```http
DELETE /v1/classroom/courses/aliases/delete
Host: api.example.com
Authorization: Bearer <your_access_token>
Content-Type: application/json

{
  "course_id": "12345",
  "alias": "old-alias"
}
```

Common Parameters:

* **course_id** (string, required): The unique identifier for the course from which the alias is to be deleted.
* **alias** (string, required): The alias that needs to be removed from the course.

Google API Documentation

Return only the requested content without any greetings or extra commentary.

Example Code

```python
import requests

def delete_course_alias(course_id, alias):
    """
    Deletes a specific alias for a course.

    :param course_id: The ID of the course.
    :param alias: The alias to delete.
    :return: Response from the API.
    """
    url = "https://classroom.googleapis.com/v1/courses/{}/aliases/{}".format(course_id, alias)
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your OAuth 2.0 access token
    }

    try:
        response = requests.delete(url, headers=headers)
        response.raise_for_status()  # Raises 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}")  # Handle HTTP errors
    except Exception as err:
        print(f"An error occurred: {err}")  # Handle any other errors
    return None

# Example usage:
course_id = "1234567890"
alias = "math101"
delete_course_alias(course_id, alias)
```

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).

Example Code

```python
import requests

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

# Define the payload with realistic parameter names and values
payload = {
    "alias": "math101",
    "courseId": "1234567890"
}

# Make the POST request to create a new course alias
try:
    response = requests.post(url, headers=headers, json=payload)
    response.raise_for_status()  # Raise an exception for HTTP errors
    data = response.json()
    print("Alias created successfully:", data)
except requests.exceptions.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.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

Endpoint: GET classroom.courses.aliases.list

Common Use Cases:
- List all alias names for courses in a classroom.
- Retrieve alias names to identify courses for administrative purposes.
- Get aliases for courses to update or manage course records.

Example Request:
```
GET /v1/classroom.courses.aliases.list?courseId=12345
```

Common Parameters:
- `courseId` (required): The unique identifier for the course.
- `maxResults` (optional): The maximum number of alias names to return. If not specified, a default value is used.
- `pageToken` (optional): A token to retrieve the next set of results, used for pagination.

Example Code

```python
import requests

def list_course_aliases(course_id, page_token=None):
    """
    Lists the aliases for a course.

    Args:
    course_id (str): The ID of the course.
    page_token (str, optional): A token identifying a page of results the server should return.
                                    If empty or not provided, the first page of results is returned.

    Returns:
    dict: The response from the API containing the aliases.
    """

    # Define the API endpoint
    url = "https://classroom.googleapis.com/v1/courses/{course_id}/aliases".format(course_id=course_id)
```

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

    # Define the parameters
    params = {}
    if page_token:
        params['pageToken'] = page_token

    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()

        # 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"An error occurred: {err}")

# Example usage
course_id = "1234567890"
aliases = list_course_aliases(course_id)
if aliases:
    print(aliases)
```

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

Endpoint: GET classroom.courses.posts.getAddOnContext

Common Use Cases:
- Fetching context information for add-ons within a specific course.
- Retrieving data required for integrating third-party tools or plugins into a course.
- Populating UI elements with relevant add-on context data.

Example Request:
```
GET https://api.example.com/classroom/courses/123/posts/getAddOnContext?courseId=123
```

Common Parameters:
- `courseId` (required): The unique identifier for the course.
- `addOnId` (optional): The unique identifier for the specific add-on, if applicable.
- `userId` (optional): The unique identifier for the user making the request, for personalized context.
- `postId` (optional): The unique identifier for the specific post, if context is needed for a particular post within the course.

Example Code

```python
import requests

def get_add_on_context(course_id, user_id, access_token):
    """
    Function to get add-on context for a course in a classroom.

    :param course_id: The ID of the course.
    :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/courses/{}/posts/addOnContext".format(course_id)
    headers = {
```

```
        "Authorization": "Bearer {}".format(access_token),
        "Content-Type": "application/json"
    }
    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}")
    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:
# course_id = "1234567890"
# user_id = "1234567890"
# access_token = "YOUR_ACCESS_TOKEN"
# add_on_context = get_add_on_context(course_id, user_id, access_token)
# print(add_on_context)
```

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

Technical Description for API Endpoint: GET classroom.courses.posts.addOnAttachments.get

Common Use Cases:
- Retrieve a list of attachments for a specific post within a course.
- Verify the presence and details of attachments for a given post.
- Integrate with applications that require access to post attachments for further processing or display.

Example Request:
```
GET /classroom/courses/{courseId}/posts/{postId}/attachments
```
Placeholder Values:
- `courseId`: ID of the course (e.g., `12345`)
- `postId`: ID of the post (e.g., `67890`)

Common Parameters:
- **courseId** (Path Parameter): The unique identifier of the course.
- **postId** (Path Parameter): The unique identifier of the post within the course.
- **filter** (Query Parameter, optional): Optional filtering criteria for attachments (e.g., file type, size).
- **limit** (Query Parameter, optional): Maximum number of attachments to return. Default is typically 10.
- **offset** (Query Parameter, optional): Number of attachments to skip. Used for pagination.

Note: The actual format of URL parameters may vary based on API design specifications.

Example Code

```python
import requests

def add_classroom_post_attachment(course_id, post_id, file_path):
    """
    Adds an attachment to a post in a Google Classroom course.

    :param course_id: ID of the course.
    :param post_id: ID of the post.
    :param file_path: Path to the file to be attached.
    :return: Response from the API.
```

```
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments"
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',  # Replace with your actual access token
        'Content-Type': 'application/json'
    }

    # Prepare the file to be uploaded
    with open(file_path, 'rb') as file:
        files = {'file': file}

    try:
        response = requests.post(url, headers=headers, files=files)
        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 Exception as err:
        print(f'Other error occurred: {err}')

# Example usage
course_id = "1234567890"
post_id = "0987654321"
file_path = "path/to/your/file.pdf"
response = add_classroom_post_attachment(course_id, post_id, file_path)
print(response)
```

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
- Uploading additional files to a course post.
- Attaching documents, images, or other files to enhance course content.
- Providing supplementary materials for course assignments.

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

{
  "course_id": "12345",
  "post_id": "67890",
  "file": [file1, file2]
}
```

Common Parameters
- **course_id** (String): The unique identifier for the course.
- **post_id** (String): The unique identifier for the course post.
- **file** (File): The file(s) to be attached to the post.

Example Code

```python
import requests
import json

# Define the endpoint and API key
url = "https://api.example.com/classroom/courses/posts/addOnAttachments/create"
api_key = "your_api_key_here"

# Define the headers
headers = {
    "Authorization": f"Bearer {api_key}",
```

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

# Define the payload with realistic parameter names and values
payload = {
    "courseId": "12345",
    "postId": "67890",
    "attachmentType": "file",
    "attachmentUrl": "https://example.com/file.txt",
    "attachmentName": "file.txt",
    "description": "Sample attachment description"
}

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

    # Check if the request was successful
    if response.status_code == 200:
        print("Attachment created successfully")
        print(response.json())
    else:
        # Print the error message if the request failed
        print(f"Error: {response.status_code}")
        print(response.json())

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

This Python script sends a POST request to the specified API endpoint to create an attachment for a post in a course. It includes proper error handling to manage potential issues that might occur during the request. The payload contains realistic parameter names and values relevant to the operation.

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 classroom course.
- Monitor and review attachments for a particular post to ensure compliance with educational standards or to manage resources.
- Integrate classroom management systems that need to display or manage attachments dynamically.

Example Request:
```http
GET https://api.example.com/classroom/courses/123/posts/456/addOnAttachments.list
```

Common Parameters:
- **courseId**: The unique identifier for the course within the classroom system.
- **postId**: The unique identifier for the post within the course.
- **type**: (Optional) Specify the type of attachments to filter by (e.g., document, image, video).
- **limit**: (Optional) The maximum number of attachments to return in the response.
- **offset**: (Optional) The number of attachments to skip before starting to return the results.

Example Code

```python
import requests

def list_attachments(service_name, course_id, post_id, api_key):
    # Define the endpoint URL
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments"

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

    # Set up the parameters for the request
    params = {
```

```
        'pageSize': 10  # Example parameter with a realistic value
    }

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

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

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

        # Return the list of attachments
        return attachments

    except requests.exceptions.HTTPError as http_err:
        print(f'HTTP error occurred: {http_err}')  # HTTP error handling
    except Exception as err:
        print(f'Other error occurred: {err}')  # General error handling
    return None

# Example usage
api_key = 'your_api_key_here'
course_id = 'course_id_example'
post_id = 'post_id_example'
attachments = list_attachments('classroom.courses.posts.addOnAttachments.list', course_id, post_id, api_key)
if attachments:
    print(attachments)
```

This code defines a function `list_attachments` that:
1. Constructs the API endpoint URL using the provided course ID and post ID.
2. Sets up headers for authorization using the provided API key.
3. Configures parameters for the request, such as `pageSize`.
4. Sends a GET request to the API and handles potential HTTP and general exceptions.
5. Returns the list of attachments if the request is successful.

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

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

Common Use Cases:
- Removing a specific attachment from a post within a course.
- Cleaning up unnecessary or outdated attachments from a post.
- Managing storage by deleting large or redundant attachments.

Example Request:
```http
DELETE /classroom/courses/123456/posts/789012/addOnAttachments/345678
```

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

Example Code

```python
import requests

def delete_post_attachments(course_id, post_id, attachment_ids):
    """
    Deletes specified attachments from a post in a course.

    :param course_id: The ID of the course.
    :param post_id: The ID of the post.
    :param attachment_ids: List of attachment IDs to delete.
    :return: Response from the API.
    """
    url = f"https://api.example.com/classroom/courses/{course_id}/posts/{post_id}/attachments"
    headers = {
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN',
        'Content-Type': 'application/json'
    }
```

```
    payload = {
        'attachmentIds': attachment_ids
    }

    try:
        response = requests.delete(url, headers=headers, json=payload)
        response.raise_for_status()  # Raise an exception for 4xx/5xx errors
        return response.json()  # Assuming the API returns 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}')

# Example usage
course_id = 12345
post_id = 67890
attachment_ids = [101, 102, 103]

result = delete_post_attachments(course_id, post_id, attachment_ids)
print(result)
```

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:
- Updating the attachment details for a specific post within a classroom course.
- Adding new attachments to an existing post.
- Modifying metadata of attachments associated with a post.

Example Request:
```http
PATCH /classroom/courses/123/posts/456/addOnAttachments HTTP/1.1
Host: example.com
Content-Type: application/json

{
  "attachments": [
    {
      "id": "attachment1",
      "url": "http://example.com/uploads/attachment1.pdf",
      "name": "Updated Report.pdf"
    }
  ]
}
```

Common Parameters:
- **attachments**: An array of objects, each containing details about the attachment to be updated or added.
  - **id**: Unique identifier for the attachment (optional if creating a new one).
  - **url**: URL of the attachment.
  - **name**: Name of the attachment file.
- **courseId**: Identifier for the course within the classroom (Path Parameter).
- **postId**: Identifier for the post within the course (Path Parameter).

Example Code

```python
import requests
```

```
def add_attachments_to_post(course_id, post_id, file_path, access_token):
    # Define the API endpoint
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments:attach"

    # Define the headers including the authorization token
    headers = {
        'Authorization': f'Bearer {access_token}',
        'Content-Type': 'multipart/form-data'
    }

    # Define the files to be uploaded
    files = {
        'attachments': (file_path, open(file_path, 'rb'))
    }

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

        # Check for HTTP errors
        response.raise_for_status()

        # Print the response from the API
        print("Attachments added successfully:", 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"
post_id = "9876543210"
file_path = "path/to/your/file.txt"
access_token = "your_access_token_here"

add_attachments_to_post(course_id, post_id, file_path, access_token)
```

This Python script defines a function `add_attachments_to_post` that takes in the course ID, post ID, file path, and access token. It constructs the API endpoint URL, sets up the headers including the authorization token, and prepares the file to be uploaded. The script then makes a POST request to add the attachment to the post and includes error handling for HTTP errors and other exceptions.

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:
- Update the status of student submissions for a specific post.
- Modify metadata of student-submitted attachments for a post.
- Add comments or feedback on student submissions.

Example Request:
```json
PATCH /classroom/courses/123/posts/456/addOnAttachments/studentSubmissions/789
Content-Type: application/json

{
  "status": "reviewed",
  "feedback": "Great work!",
  "comments": "Consider adding more details in the next submission."
}
```

Common Parameters:
- `status` (optional): The new status of the student submission (e.g., "submitted", "reviewed", "approved").
- `feedback` (optional): Additional feedback or comments from the instructor.
- `comments` (optional): Detailed comments or notes related to the submission.
- `courseId` (path parameter): The unique identifier for the course.
- `postId` (path parameter): The unique identifier for the post within the course.
- `attachmentId` (path parameter): The unique identifier for the attachment associated with the student submission.
- `submissionId` (path parameter): The unique identifier for the student submission.

Example Code

```python
import requests

# Define the endpoint URL and the API key
```

```
url = "https://classroom.googleapis.com/v1/courses/{course_id}/posts/{post_id}/attachments/studentSubmissions/{student_id}"
api_key = "YOUR_API_KEY"
course_id = "1234567890"
post_id = "1234567890"
student_id = "student123"

# Define the patch payload
patch_payload = {
    "attachments": [
        {
            "mimeType": "application/pdf",
            "title": "Updated Assignment",
            "sourceUrl": "https://example.com/updated-assignment.pdf",
        }
    ]
}

# Set up the headers for authentication
headers = {
    "Authorization": f"Bearer {api_key}",
    "Content-Type": "application/json"
}

# Make the PATCH request to the API endpoint
response = requests.patch(
    url.format(course_id=course_id, post_id=post_id, student_id=student_id),
    headers=headers,
    json=patch_payload
)

# Check for HTTP errors
if response.status_code == 200:
    print("Attachment updated successfully.")
else:
    print(f"Failed to update attachment. Status code: {response.status_code}")
    print(f"Response: {response.json()}")
```

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

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

Common Use Cases:
- Retrieve attachments submitted by students for a specific assignment.
- Verify which students have completed and attached their work.
- Audit or review student submissions for quality assurance.

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

Common Parameters:
- `courseId` (string, required): The unique identifier for the course.
- `postId` (string, required): The unique identifier for the assignment post.
- `access_token` (string, required): The authentication token for API access.

Example Code

```python
import requests

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

# Define the parameters for the request
course_id = "1234567890"
post_id = "0987654321"
attachment_id = "1234567890"
student_id = "exampleStudentId"

# Construct the full URL with the parameters
full_url = url.format(course_id=course_id, post_id=post_id, attachment_id=attachment_id)
```

```
# Define the headers, including the authorization token
headers = {
    "Authorization": "Bearer YOUR_ACCESS_TOKEN"
}

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

# Check for HTTP errors
if response.status_code != 200:
    # Print the error message if the request was not successful
    print(f"Error: {response.status_code}")
    print(f"Message: {response.json().get('error', {}).get('message', 'Unknown error')}")
else:
    # Parse the JSON response
    student_submissions = response.json()

    # Print the student submissions
    print("Student Submissions:")
    for submission in student_submissions.get('studentSubmissions', []):
        print(f"Student ID: {submission.get('studentId')}, Submission ID: {submission.get('id')}")
```

Explanation:
1. **URL Construction**: The URL is constructed using the provided parameters (`course_id`, `post_id`, `attachment_id`).
2. **Headers**: The `Authorization` header includes a bearer token for authentication.
3. **GET Request**: A GET request is made to the constructed URL.
4. **Error Handling**: If the response status code is not 200 (OK), an error message is printed.
5. **Response Parsing**: If the request is successful, the JSON response is parsed and printed.

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

Technical Description:
Endpoint: PATCH classroom.courses.courseWorkMaterials.patch

Common Use Cases:
- Update the status of coursework materials.
- Modify the details of existing coursework materials.
- Update the due date or associated resources of coursework materials.

Example Request:
```json
PATCH /v1/classroom/courses/{courseId}/courseWorkMaterials/{courseWorkMaterialId}
Content-Type: application/json

{
  "status": "completed",
  "dueDate": "2023-12-31",
  "resources": [
    {
      "name": "Updated Resource",
      "url": "https://example.com/new-resource"
    }
  ]
}
```

Common Parameters:
- **courseId** (Path Parameter): The unique identifier of the course.
- **courseWorkMaterialId** (Path Parameter): The unique identifier of the coursework material.
- **status** (Query Parameter): The status of the coursework material (e.g., "pending", "completed").
- **dueDate** (Query Parameter): The new due date for the coursework material (format: YYYY-MM-DD).
- **resources** (Query Parameter): An array of resource objects containing the name and URL of the resources.

Example Code

```python
import requests
```

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

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

# Define the parameters for the request
params = {
    "courseId": "1234567890",
    "courseWorkId": "12345",
    "materialId": "67890"
}

# Define the data to be sent in the request body
data = {
    "title": "Updated Lesson Title",
    "description": "Updated Lesson Description",
    "materialType": "driveFile",
    "link": "https://drive.google.com/file/d/1234567890/view",
    "alternateLink": "https://drive.google.com/file/d/1234567890"
}

# Include error handling
try:
    # Send the PATCH request to update the course work material
    response = requests.patch(url.format(**params), headers=headers, json=data)

    # Check for successful response
    response.raise_for_status()

    # Print the updated course work material details
    print(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}")
```

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

Technical Description for POST classroom.courses.courseWorkMaterials.create

Common Use Cases:
- **Teacher:** Upload course materials such as lecture slides, assignments, or readings for students to access.
- **Admin:** Distribute standardized materials across multiple courses.
- **Student:** Access and download course materials provided by the instructor.

Example Request:
```json
POST https://api.example.com/classroom/courses/{courseId}/courseWorkMaterials
{
  "title": "Lecture Slides Week 1",
  "file": {
    "name": "Week1_Slides.pdf",
    "data": "base64_encoded_file_data"
  },
  "description": "Slides for the first week's lecture on Introduction to Computer Science."
}
```

Common Parameters:
- `courseId` (path): The unique identifier for the course.
- `title` (body): The title of the course material.
- `file` (body): An object containing the file details.
  - `name` (body): The name of the file to be uploaded.
  - `data` (body): The base64 encoded data of the file.
- `description` (body): A brief description of the course material.

Example Code

```python
import requests

# Define the API endpoint and authentication details
url = "https://classroom.googleapis.com/v1/courses/{course_id}/courseWorkMaterials"
```

```
auth_token = "YOUR_ACCESS_TOKEN"  # Replace with your actual access token

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

# Define the parameters for the request
data = {
    "title": "Introduction to Python",
    "driveFile": {
        "driveFileId": "1a2b3c4d5e6f7g8h9i0j"
    },
    "alternateLink": "https://drive.google.com/file/d/1a2b3c4d5e6f7g8h9i0j/view?usp=sharing"
}

# Define the course ID
course_id = "1234567890"  # Replace with your actual course ID

# Make the API request
try:
    response = requests.post(f"{url}/{course_id}", headers=headers, json=data)
    response.raise_for_status()  # Raise an exception for HTTP errors
    print("Course work material created successfully:", 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}")
```

This code creates a course work material for a specific course in Google Classroom using the `classroom.courses.courseWorkMaterials.create` endpoint. It includes proper error handling to manage potential issues with the API request.

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

Common Use Cases:
- **Admin Deletion**: An administrator removes outdated course materials to keep the course content up-to-date.
- **Student Request**: A student may request the deletion of incorrect or irrelevant course materials.
- **Instructor Adjustment**: An instructor updates the course materials by deleting unnecessary resources.

Example Request:
```
DELETE /classroom.courses.courseWorkMaterials.delete
Authorization: Bearer YOUR_ACCESS_TOKEN
Content-Type: application/json

{
  "course_id": "abcd1234",
  "material_id": "5678efgh"
}
```

Common Parameters:
- `course_id`: The unique identifier of the course from which the material is to be deleted.
- `material_id`: The unique identifier of the course material to be deleted.
- `Authorization`: The bearer token for authentication.
- `Content-Type`: Specifies the media type of the request body.

Example Code

```python
import requests

def delete_course_work_material(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 for authentication
    headers = {
        'Authorization': f'Bearer {api_key}'
    }

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

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

        # If successful, the material should be deleted
        print("Course work material deleted successfully.")
        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 any other exceptions
        print(f"An error occurred: {err}")

# Example usage
course_id = '1234567890'
course_work_id = 'abcdef1234567890'
api_key = 'your_api_key_here'

delete_course_work_material(course_id, course_work_id, api_key)
```

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

Technical Description for GET classroom.courses.courseWorkMaterials.get

Common Use Cases
- Retrieve course materials for a specific course.
- Access assignments, readings, and other resources.
- Integrate with learning management systems for educational purposes.

Example Request
```http
GET https://classroom.googleapis.com/v1/courses/{courseId}/courseWorkMaterials
Authorization: Bearer {accessToken}
```

Common Parameters
- **courseId** (string, required): The identifier for the course. This is typically a numeric or alphanumeric ID provided by the Google Classroom API.
- **accessToken** (string, required): The OAuth 2.0 token used to authorize the request. This token must have appropriate scopes for accessing classroom data.
- **pageSize** (integer, optional): The maximum number of results to return per page. Default is 100.
- **pageToken** (string, optional): A token identifying a page of results to return. This is used for pagination.

Example Code

```python
import requests

def get_course_work_materials(course_id, material_id):
    """
    Fetch course work materials using the classroom.courses.courseWorkMaterials.get endpoint.

    :param course_id: The ID of the course to get course work materials for.
    :param material_id: The ID of the course work material to get.
    :return: The course work material data if successful, otherwise None.
    """
    url = f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWorkMaterials/{material_id}"
    headers = {
```

```
        'Authorization': 'Bearer YOUR_ACCESS_TOKEN'  # Replace with your actual access token
    }

    try:
        response = requests.get(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 requests.exceptions.RequestException as err:
        print(f"Error occurred: {err}")
    return None

# Example usage
course_materials = get_course_work_materials('course12345', 'material67890')
if course_materials:
    print("Course work materials:", course_materials)
else:
    print("Failed to retrieve course work 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

Technical Description

Endpoint: GET classroom.courses.courseWorkMaterials.getAddOnContext

Common Use Cases:
1. Retrieve add-on context for coursework materials in Google Classroom.
2. Integrate third-party applications with Google Classroom to provide additional resources or tools.
3. Facilitate educational analytics and reporting by accessing contextual information about coursework materials.

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

Common Parameters:
1. **courseId (Path Parameter):** The unique identifier for the course. Required.
2. **courseWorkMaterialId (Path Parameter):** The unique identifier for the coursework material. Required.
3. **accessToken (Query Parameter):** The OAuth 2.0 access token for authentication. Required.

Example Code

```python
import requests

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

# Define the parameters for the request
params = {
```

```
    "courseId": "1234567890",  # Replace with a valid course ID
    "courseWorkMaterialId": "9876543210"  # Replace with a valid course work material ID
}

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

# Send the GET request to the API endpoint
try:
    response = requests.get(url, headers=headers, params=params)
    response.raise_for_status()  # Raise an HTTPError for bad responses (4xx or 5xx)
    data = response.json()
    # Process the response data as needed
    print(data)
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except requests.exceptions.RequestException as err:
    print(f"An error occurred: {err}")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

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

Endpoint: GET classroom.courses.courseWorkMaterials.list

Common Use Cases:
- Retrieve a list of coursework materials for a specific course.
- Fetch educational resources assigned by the instructor.
- Access different types of coursework materials such as documents, videos, and links.

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

Common Parameters:
- **courseId**: The identifier for the course whose materials are being listed.
- **pageSize**: (Optional) The maximum number of materials to return per page. The default is 20.
- **pageToken**: (Optional) The token identifying the page of results to return. If empty, the first page of results will be returned.

Example Code

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

def list_course_work_materials(course_id):
    # Authenticate and build the service
    credentials, _ = google.auth.default()
    service = build('classroom', 'v1', credentials=credentials)

    try:
        # Define the parameters for the request
        params = {
            'courseId': course_id  # Replace with the actual course ID
        }
```

```
        # Call the API endpoint
        course_work_materials = service.courses().courseWorkMaterials().list(**params).execute()

        # Print the returned course work materials
        return course_work_materials
    except googleapiclient.errors.HttpError as e:
        # Handle errors
        print(f'An error occurred: {e}')
        return None

# Example usage
course_id = '1234567890'
course_work_materials = list_course_work_materials(course_id)
if course_work_materials:
    print(course_work_materials)
```

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
1. Retrieve metadata and details of attachments linked to coursework materials.
2. Verify the existence and status of attachments for specific coursework.
3. Integrate with course management systems to manage and display attachments.

Example Request
```http
GET
https://classroom.googleapis.com/v1/classroom.courses.courseWorkMaterials.addOnAttachments/get?courseId=your_course_id&courseWorkId=your_coursework_id&attachmentId=your_attachment_id
```

Common Parameters
1. **courseId** (string, required): The ID of the course.
2. **courseWorkId** (string, required): The ID of the coursework.
3. **attachmentId** (string, required): The ID of the attachment.

The endpoint returns details about the specified attachment linked to the coursework materials within a course.

Example Code

```python
import requests

# Define the API endpoint and parameters
url = "https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/attachments"
course_id = "1234567890"
course_work_id = "9876543210"
material_id = "5555555555"
access_token = "YOUR_ACCESS_TOKEN"

# Set up the headers with the access token
headers = {
```

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

# Make the API request
try:
    response = requests.get(url, headers=headers)
    response.raise_for_status()  # Raise an error for bad status codes
    attachments = response.json()  # Parse the JSON response
    print(attachments)  # Print the attachments data
except requests.exceptions.HTTPError as http_err:
    print(f"HTTP error occurred: {http_err}")
except Exception as err:
    print(f"An error occurred: {err}")
```

This code snippet demonstrates how to make a GET request to the `classroom.courses.courseWorkMaterials.addOnAttachments.get` API endpoint, using realistic parameter names and values. It includes proper error handling to catch and print any errors that occur during the request.

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

Common Use Cases:
- Updating existing attachments for course materials.
- Adding new attachments to existing course materials.
- Modifying metadata of attachments without removing them.

Example Request:
```http
PATCH /classroom.courses.courseWorkMaterials.addOnAttachments.patch
Content-Type: application/json

{
  "courseWorkId": "12345",
  "materialId": "67890",
  "attachments": [
    {
      "attachmentId": "11111",
      "newTitle": "Updated Course Plan",
      "newDescription": "This is the updated course plan for the semester."
    },
    {
      "newAttachment": {
        "title": "New Lecture Notes",
        "description": "Lecture notes for the current week.",
        "fileUrl": "http://example.com/lecture-notes.pdf"
      }
    }
  ]
}
```

Common Parameters:
- **courseWorkId** (string): The ID of the course work to which the attachments belong.
- **materialId** (string): The ID of the specific course material being updated.
- **attachments** (array):
  - **newAttachment** (object): Contains details for a new attachment being added.
    - **title** (string): The title of the new attachment.
    - **description** (string): A description of the new attachment.

- **fileUrl** (string): The URL of the file to be attached.
- **attachmentId** (string): The ID of an existing attachment being updated.
- **newTitle** (string): The new title for the existing attachment.
- **newDescription** (string): The new description for the existing attachment.

Example Code

```python
import requests

def patch_course_work_materials(id, course_id, material_id, attachments):
    # URL for the API endpoint
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{id}/materials/{material_id}/attachments"

    # Headers for the request, including authorization
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }

    # Payload with the attachments data
    payload = {
        "attachments": attachments
    }

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

        # 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:
        # Handle HTTP errors
        print(f"HTTP error occurred: {http_err}")
    except Exception as err:
        # Handle other errors
        print(f"An error occurred: {err}")

# Example usage
course_id = "1234567890"
id = "course_work_id"
material_id = "material_id"
attachments = [
    {
```

```
        "url": "https://example.com/attachment1.pdf",
        "title": "Attachment 1"
    },
    {
        "url": "https://example.com/attachment2.pdf",
        "title": "Attachment 2"
    }
]

result = patch_course_work_materials(id, course_id, material_id, attachments)
if result:
    print("Attachments updated successfully:", result)
```

Ensure to replace `"YOUR_ACCESS_TOKEN"` with a valid OAuth 2.0 token. The example parameters and values are realistic and can be adjusted as per the actual requirements.

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

API Endpoint: GET classroom.courses.courseWorkMaterials.addOnAttachments.list

Common Use Cases
- Retrieve a list of attachments for a specific coursework material.
- Verify the presence and details of all attachments related to a coursework material.
- Check if any attachments have been added or modified recently.

Example Request
```
GET
https://classroom.googleapis.com/v1/courses/{courseId}/courseWork/{courseWorkId}/materials/{materialId}/addOnAttachments
Authorization: Bearer {accessToken}
```

Common Parameters
- **courseId** (string, required): The ID of the course.
- **courseWorkId** (string, required): The ID of the coursework material.
- **materialId** (string, required): The ID of the material within the coursework.
- **accessToken** (string, required): The OAuth 2.0 access token for authorization.

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

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

Common Use Cases:
- Removing a specific attachment from a coursework material.
- Cleaning up obsolete or incorrect attachments.
- Managing course materials by deleting unnecessary files.

Example Request:
```http
DELETE
/classroom/courses/{courseId}/courseWorkMaterials/{courseWorkMaterialId}/addOnAttachments/{attachmentId}
```

Common Parameters:
- **courseId** (Path Parameter): The unique identifier for the course.
- **courseWorkMaterialId** (Path Parameter): The unique identifier for the coursework material.
- **attachmentId** (Path Parameter): The unique identifier for the attachment to be deleted.

Example Code

```python
import requests

def delete_course_work_material_attachment(course_id, course_work_id, attachment_id, access_token):
    """
    Deletes an attachment from a course work material.

    :param course_id: The ID of the course.
    :param course_work_id: The ID of the course work.
    :param attachment_id: The ID of the attachment to delete.
    :param access_token: The access token for authentication.
    """
                                                                                    url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/materials/attachments/{at
tachment_id}"

    headers = {
```

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

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

        # Check if the request was successful
        if response.status_code == 204:
            print("Attachment deleted successfully.")
        else:
            print(f"Failed to delete attachment. Status code: {response.status_code}")
            print(f"Response: {response.json()}")

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

# Example usage
course_id = '12345678901234567890'
course_work_id = '12345678901234567890'
attachment_id = '12345678901234567890'
access_token = 'YOUR_ACCESS_TOKEN'

delete_course_work_material_attachment(course_id, course_work_id, attachment_id, access_token)
```

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 API Endpoint: POST classroom.courses.courseWorkMaterials.addOnAttachments.create

Common Use Cases:
- Uploading supplementary materials for a coursework assignment.
- Adding reference documents or additional resources to an existing assignment.
- Enabling students to download extra materials that support the coursework.

Example Request:
```http
POST /classroom/courses/{courseId}/courseWorkMaterials/{courseWorkMaterialId}/addOnAttachments/create
Headers:
  Authorization: Bearer {access_token}
  Content-Type: multipart/form-data

Body:
  file: {file_to_upload}
```

Common Parameters:
- **courseId** (Path): Unique identifier of the course.
- **courseWorkMaterialId** (Path): Unique identifier of the coursework material.
- **Authorization** (Header): Bearer token for authentication.
- **Content-Type** (Header): Specifies the media type of the resource. For file uploads, it should be `multipart/form-data`.
- **file** (Form Data): The file to be attached to the coursework material.

Example Code

```python
import requests

def add_course_work_materials_attachment(course_id, course_work_id, file_id):
    """
                    Adds an attachment to course work materials using the
classroom.courses.courseWorkMaterials.addOnAttachments.create API.

    :param course_id: The ID of the course.
```

```
    :param course_work_id: The ID of the course work.
    :param file_id: The ID of the attachment file.
    :return: The response from the API.
    """
                                                                                url                     =
f"https://classroom.googleapis.com/v1/courses/{course_id}/courseWork/{course_work_id}/materials/addOnAttachment
s"

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

    payload = {
        "attachmentId": file_id
    }

    try:
        response = requests.post(url, headers=headers, json=payload)
        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}")  # Handle HTTP errors
    except Exception as err:
        print(f"An error occurred: {err}")  # Handle other exceptions

# Example usage
course_id = "course_123"
course_work_id = "course_work_456"
file_id = "file_789"

add_course_work_materials_attachment(course_id, course_work_id, file_id)
```

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

API Endpoint: GET classroom.userProfiles.get

Common Use Cases:
- Fetching user profiles for a specific classroom.
- Retrieving user details for administrative purposes.
- Allowing teachers to view student profiles.

Example Request:
```
GET /classroom/userProfiles/get?classroomId=12345
```

Common Parameters:
- **classroomId**: (Required) The unique identifier for the classroom whose user profiles are to be retrieved.
- **userId**: (Optional) The unique identifier for a specific user whose profile is to be retrieved. If provided, only the profile of the specified user will be returned.
- **role**: (Optional) Filter users by their role in the classroom (e.g., 'student', 'teacher', 'admin').
- **limit**: (Optional) The maximum number of user profiles to return in the response.
- **offset**: (Optional) The number of user profiles to skip before starting to collect the result set.

Example Code

```python
import requests

def get_user_profiles(api_key, user_ids):
    """
    Fetch user profiles using the classroom.userProfiles.get API endpoint.

    Args:
    api_key (str): The API key for authentication.
    user_ids (list): A list of user IDs to fetch profiles for.

    Returns:
    dict: A dictionary containing the user profiles.

    Raises:
    Exception: If the API request fails.
```

```
    """
    url = "https://classroom.googleapis.com/v1/userProfiles"
    headers = {
        "Authorization": f"Bearer {api_key}"
    }
    params = {
        "userIds": ",".join(user_ids)  # Join user IDs as a comma-separated string
    }

    try:
        response = requests.get(url, headers=headers, params=params)
        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 Exception as err:
        print(f"Other error occurred: {err}")
    return None

# Example usage:
api_key = "your_api_key_here"
user_ids = ["user1@example.com", "user2@example.com"]

user_profiles = get_user_profiles(api_key, user_ids)
if user_profiles:
    print("User Profiles:", user_profiles)
else:
    print("Failed to retrieve user profiles.")
```

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

API Endpoint: POST classroom.userProfiles.guardianInvitations.create

Common Use Cases:
- Invite a guardian to monitor a student's classroom activities.
- Send an invitation to parents or guardians to be involved in their child's educational progress.
- Allow teachers to manage guardian permissions and notifications for student profiles.

Example Request:
```http
POST /v1/classroom/userProfiles/guardianInvitations/create
Content-Type: application/json
Authorization: Bearer YOUR_ACCESS_TOKEN
{
  "studentId": "12345",
  "inviteeEmail": "guardian@example.com",
  "inviterId": "67890"
}
```

Common Parameters:
- **studentId** (string): The unique identifier for the student to whom the guardian invitation is being sent.
- **inviteeEmail** (string): The email address of the guardian being invited.
- **inviterId** (string): The unique identifier of the user (e.g., teacher) who is sending the invitation.
- **customMessage** (optional, string): An optional custom message to include with the invitation.

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

Technical Description

Common Use Cases
- Retrieve a list of guardian invitations for a specific user profile within a classroom.
- Monitor and manage guardian invitations to ensure proper access and permissions are granted.
- Generate reports on guardian invitations for administrative purposes.

Example Request
```
GET /classroom/userProfiles/{userProfileId}/guardianInvitations
```

Common Parameters
- **userProfileId**: (Path Parameter) The unique identifier for the user profile. Required.
- **access_token**: (Query Parameter) The access token for authorization. Required.
- **page_token**: (Query Parameter) The token for the next page of results. Optional.
- **page_size**: (Query Parameter) The number of results to return per page. Optional.

Example Code

```python
import requests
import json

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

# Define the parameters for the API request
```

```
params = {
    "pageSize": 10,  # Number of items to retrieve per page
    "pageToken": "abc123"  # Token to retrieve the next set of results
}

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

# Check for HTTP errors
if response.status_code != 200:
    # Print the error message if any
    print(f"Error: {response.status_code}")
    print(response.json())  # Print the error details
else:
    # Parse the JSON response
    data = response.json()

    # Print the returned data
    print(json.dumps(data, indent=2))

# Handle specific cases where the response might not contain the expected data
try:
    guardian_invitations = data.get('guardianInvitations', [])
    for invitation in guardian_invitations:
        print(f"Invitation ID: {invitation['id']}")
        print(f"Email: {invitation['inviteeEmail']}")
        print(f"Role: {invitation['role']}")
        print(f"State: {invitation['state']}")
except KeyError as e:
    print(f"Key error: {e}")

# Note: Replace 'YOUR_ACCESS_TOKEN' with a valid OAuth 2.0 access token.
# The 'pageToken' parameter should be set based on the previous response to handle pagination.
```

This script sends a GET request to the `classroom.userProfiles.guardianInvitations.list` endpoint, includes proper error handling, and processes the JSON response to extract and print specific details about guardian invitations.

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 for GET classroom.userProfiles.guardianInvitations.get

Common Use Cases:
1. Retrieve a list of guardian invitations sent to specific guardians for a particular classroom.
2. Verify the status of guardian invitations to ensure they have been accepted or need to be resent.

Example Request:
```
GET /v1/classroom.userProfiles.guardianInvitations.get?classroomId=123456&guardianEmail=guardian@example.com
```

Common Parameters:
1. **classroomId** (required): The unique identifier for the classroom.
2. **guardianEmail** (required): The email address of the guardian whose invitations are being queried.
3. **status** (optional): Filter invitations by status (e.g., 'pending', 'accepted', 'rejected').

Example Code

```python
import requests

def get_guardian_invitations(service_id, student_id, token):
    """
    Function to fetch guardian invitations for a specific student in a classroom.

    :param service_id: The ID of the service.
    :param student_id: The ID of the student.
    :param token: The authentication token.
    :return: JSON response from the API.
    """
    url = "https://classroom.googleapis.com/v1/guardianInvitations"
    headers = {
        "Authorization": f"Bearer {token}"
```

```
    }
    params = {
        "serviceId": service_id,
        "studentId": student_id
    }

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

        # 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 Exception as err:
        print(f"An error occurred: {err}")

    return None

# Example usage:
service_id = "example-service-id"  # Replace with a real service ID
student_id = "student-12345"       # Replace with a real student ID
token = "your-auth-token"           # Replace with a real auth token

guardian_invitations = get_guardian_invitations(service_id, student_id, token)
print(guardian_invitations)
```

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:
- Updating the invitation status for a guardian.
- Modifying the invitation details such as the email or name of the guardian.
- Revoking an invitation to a guardian.

Example Request:

```http
PATCH /v1/classrooms/{classroomId}/userProfiles/{userProfileId}/guardianInvitations/{invitationId} HTTP/1.1
Host: api.example.com
Content-Type: application/json
Authorization: Bearer YOUR_ACCESS_TOKEN

{
  "status": "pending",
  "email": "newguardian@example.com",
  "name": "Guardian Name"
}
```

Common Parameters:

- `classroomId` (URL): The unique identifier for the classroom.
- `userProfileId` (URL): The unique identifier for the user profile.
- `invitationId` (URL): The unique identifier for the guardian invitation.
- `status` (body): The status of the invitation (e.g., "pending", "accepted", "revoked").
- `email` (body): The email address of the guardian.
- `name` (body): The name of the guardian.

Example Code

```python
import requests

def patch_guardian_invitation(user_id, guardian_email, invitation_id):
    # Define the API endpoint and headers
    url = f"https://classroom.googleapis.com/v1/userProfiles/{user_id}/guardianInvitations/{invitation_id}"
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN",
        "Content-Type": "application/json"
    }

    # Define the data to be updated
    data = {
        "guardianEmail": guardian_email
    }

    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}")
    except Exception as err:
        print(f"An error occurred: {err}")

# Example usage
user_id = "1234567890"
guardian_email = "guardian@example.com"
invitation_id = "invitation123"

# Call the function and print the result
result = patch_guardian_invitation(user_id, guardian_email, invitation_id)
print(result)
```

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

API Endpoint: GET classroom.userProfiles.guardians.list

Common Use Cases
- Retrieve a list of guardians associated with a specific user profile in a classroom.
- Gather guardian information for administrative purposes.
- Facilitate communication between guardians and educators.

Example Request
```
GET /api/classroom/userProfiles/{userProfileId}/guardians
```
- **Endpoint:** /api/classroom/userProfiles/{userProfileId}/guardians
- **Method:** GET
- **Path Parameters:**
  - `userProfileId` (string): The unique identifier of the user profile for whom guardians are being listed.

Common Parameters
- **userProfileId** (Required): The unique identifier of the user profile.
- **page** (Optional): The page number to retrieve for pagination.
- **pageSize** (Optional): The number of guardians to return per page.
- **sort** (Optional): The field by which to sort the results (e.g., name, relationship).

Example Code

```python
import requests

def list_guardians(student_id, access_token):
    """
    This function lists the guardians of a student using the classroom.userProfiles.guardians.list API.
```

```
    :param student_id: The ID of the student whose guardians are to be listed.
    :param access_token: The OAuth 2.0 token used for authorization.
    :return: A list of guardians or an error message.
    """
    url = 'https://classroom.googleapis.com/v1/users/me/profile/guardians'

    # Define the query parameters
    params = {
        'studentId': student_id,
        'fields': 'guardianDetails(guardianId,relationship,emailAddress,guardianId)'
    }

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

    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
        guardians = response.json().get('guardianDetails', [])

        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}')

# Example usage
student_id = '1234567890'  # Replace with a real student ID
access_token = 'ya29...'  # Replace with a valid OAuth 2.0 token

guardians = list_guardians(student_id, access_token)
print(guardians)
```

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

Endpoint: DELETE classroom.userProfiles.guardians.delete

Common Use Cases:
- Remove a guardian from a user's profile.
- Update user records to reflect changes in guardianship.
- Ensure data accuracy by deleting outdated or incorrect guardian information.

Example Request:
```
DELETE https://api.example.com/classroom/userProfiles/12345/guardians/67890
```

Common Parameters:
- `userProfileId`: The unique identifier for the user profile from which the guardian is to be removed.
- `guardianId`: The unique identifier for the guardian to be deleted.

Example Code

```python
import requests

def delete_guardian(user_id, guardian_id, access_token):
    """
    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 be deleted.
    :param access_token: The OAuth 2.0 access token.
    :return: Response from the API.
    """

    url = "https://classroom.googleapis.com/v1/users/{}/guardians/{}".format(user_id, guardian_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)

        # Success case
        print("Guardian deleted successfully.")
        return response.json()

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

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

# Example usage
user_id = "1234567890"
guardian_id = "0987654321"
access_token = "your_access_token_here"

delete_guardian(user_id, guardian_id, access_token)
```

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

Endpoint: GET classroom.userProfiles.guardians.get

Common Use Cases:
- Retrieve a list of guardians associated with a specific user profile in a classroom.
- Access guardian information for administrative purposes, such as contact details or emergency information.
- Integrate guardian data into third-party applications for enhanced communication or tracking.

Example Request:
```
GET https://api.example.com/classroom/userProfiles/guardians?userProfileId=12345
```

Common Parameters:
- `userProfileId` (string, required): The unique identifier for the user profile whose guardians are being retrieved.
- `fields` (string, optional): Specifies a comma-separated list of fields to be included in the response. Defaults to all available fields.
- `limit` (integer, optional): The maximum number of guardians to return. Defaults to 10.
- `offset` (integer, optional): The number of guardians to skip before returning results. Used for pagination.

Return only the requested content without any greetings or extra commentary.

Example Code

```python
import requests

def get_guardians(user_id, access_token):
    url = "https://classroom.googleapis.com/v1/users/{}/guardians".format(user_id)
    headers = {
        "Authorization": f"Bearer {access_token}"
    }
```

```
    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()  # Raise an exception for HTTP errors

        # Parse the JSON response
        guardians = response.json()
        return guardians

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

# Example usage
user_id = "1234567890"  # Replace with a valid user ID
access_token = "your_access_token_here"  # Replace with a valid access 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. It makes a GET request to the `classroom.userProfiles.guardians.get` endpoint to retrieve the guardians associated with the specified user. It includes proper error handling for HTTP errors and other exceptions. The example usage at the bottom demonstrates how to call this function with realistic parameter values.

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

Technical Description for GET classroom.invitations.get

Common Use Cases:
- Retrieving a list of invitations for a specific classroom.
- Verifying if a user has received an invitation to join a classroom.
- Checking the status of pending invitations.

Example Request:
```
GET /classroom/invitations?classroom_id=12345&user_id=67890
```

Common Parameters:
- **classroom_id**: The unique identifier for the classroom whose invitations are to be retrieved.
- **user_id**: The unique identifier for the user whose invitations are to be checked.
- **status**: (Optional) The status of the invitations to filter by (e.g., pending, accepted, declined).

Example Code

```python
import requests

def get_classroom_invitation(classroom_id, user_email):
    # Define the API endpoint URL
    url = "https://classroom.googleapis.com/v1/classrooms/{}/invitations".format(classroom_id)

    # Define the headers for the request
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with your actual access token
    }

    # Define the parameters for the request
    params = {
        "userEmail": user_email
    }

    try:
```

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

        # Check if the request was successful
        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 Exception as err:
        print(f"An error occurred: {err}")

# Example usage
classroom_id = "1234567890"  # Replace with actual classroom ID
user_email = "example@example.com"  # Replace with actual user email

invitation_data = get_classroom_invitation(classroom_id, user_email)
if invitation_data:
    print(invitation_data)
```

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

Technical Description for POST classroom.invitations.create

Common Use Cases:
- Inviting new students to a classroom.
- Adding teaching assistants or co-teachers to a classroom.
- Automating the enrollment process for educational institutions.

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

{
  "classroom_id": "12345",
  "invitation_type": "student",
  "email": "student@example.com",
  "additional_info": {
    "message": "Welcome to our classroom!"
  }
}
```

Common Parameters:
- **classroom_id**: (string, required) Unique identifier for the classroom.
- **invitation_type**: (string, required) Type of invitation, e.g., "student", "teacher", "assistant".
- **email**: (string, required) Email address of the person being invited.
- **additional_info**: (object, optional) Additional information for the invitation, such as a welcome message.

Example Code

```python
import requests

def create_classroom_invitation(classroom_id, email, role):
```

```
    # Define the API endpoint URL
    url = "https://api.example.com/classroom/invitations"

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

    # Define the payload with realistic parameter names and values
    payload = {
        'classroom_id': classroom_id,
        'email': email,
        'role': role
    }

    try:
        # Make the POST request to create the invitation
        response = requests.post(url, json=payload, 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 Exception as err:
        print(f'Other error occurred: {err}')

# Example usage
classroom_id = '12345'
email = 'user@example.com'
role = 'teacher'

invitation = create_classroom_invitation(classroom_id, email, role)
if invitation:
    print('Invitation created successfully:', invitation)
```

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

The `POST classroom.invitations.accept` endpoint is used to accept an invitation to join a classroom. This endpoint is commonly used in educational platforms to manage classroom memberships.

Common Use Cases:
1. Accepting an invitation to join a classroom as a student or teacher.
2. Automatic acceptance of invitations sent by administrators or teachers.
3. Confirming participation in a specific educational course or group.

Example Request:
```json
POST /api/classroom/invitations/accept
{
  "invitationId": "12345",
  "acceptanceToken": "abcdef"
}
```

Common Parameters:
- **invitationId** (string, required): The unique identifier for the invitation being accepted.
- **acceptanceToken** (string, required): A token provided in the invitation email or message to verify the acceptance.

Example Code

```python
import requests

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

    Parameters:
    token (str): The authentication token for the API.
    invitation_id (str): The ID of the invitation to accept.
```

```
    Returns:
    dict: The response from the API.
    """
    url = "https://api.example.com/classroom/invitations/accept"  # Replace with actual URL

    headers = {
        "Authorization": f"Bearer {token}",
        "Content-Type": "application/json"
    }

    data = {
        "invitation_id": invitation_id
    }

    try:
        response = requests.post(url, headers=headers, json=data)
        response.raise_for_status()  # Raise HTTPError for bad responses (4xx or 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:
token = "your_auth_token_here"
invitation_id = "1234567890abcdef"

response = accept_classroom_invitation(token, invitation_id)
if response:
    print("Invitation accepted successfully:", response)
else:
    print("Failed to accept the invitation.")
```

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

GET classroom.invitations.list

Common Use Cases:
- Retrieve a list of pending invitations for a particular classroom.
- Check the status of invitations sent to users.
- Monitor and manage classroom memberships.

Example Request:
```
GET /classroom.invitations.list?classroomId=12345
```

Common Parameters:
- `classroomId`: The unique identifier for the classroom whose invitations are to be listed. (Required)
- `status`: Filter invitations by status (e.g., 'pending', 'accepted', 'declined'). (Optional)
- `limit`: The maximum number of invitations to return. (Optional)
- `offset`: The number of invitations to skip before starting to collect the result set. (Optional)
- `invitedUserId`: Filter invitations by the user ID of the invited person. (Optional)

Example Code

```python
import requests

def list_classroom_invitations():
    url = "https://classroom.googleapis.com/v1/invitations"
    params = {
        "pageSize": 10,  # Number of invitations to return per page
        # Add other parameters as needed, e.g., "inviterUserId", "invitationStatus", etc.
    }
    headers = {
        "Authorization": "Bearer YOUR_ACCESS_TOKEN"  # Replace with actual access token
    }

    try:
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()  # Raise an exception for HTTP errors
```

```
        invitations = response.json()  # Parse JSON response
        return invitations
    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 ValueError as val_err:
        print(f"JSON decode error: {val_err}")

# Example usage
if __name__ == "__main__":
    invitations = list_classroom_invitations()
    if invitations:
        print(invitations)
```

```python
# Dependencies and authorization token setup are assumed to be in place.
# Modify the parameters and headers as needed for your specific use case.
```

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

API Endpoint: DELETE classroom.invitations.delete

Common Use Cases:
- **Remove a Pending Invitation:** Administrators or teachers can use this endpoint to remove a pending invitation to a classroom.
- **Revoke Invitation Permissions:** Delete an invitation that was sent out to join a classroom, particularly useful if the invitation was sent in error.
- **Manage Classroom Membership:** Remove an invitation to simplify the process of classroom membership management by eliminating unnecessary pending invitations.

Example Request:
```http
DELETE /classroom/invitations/{invitationId}
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Common Parameters:
- **invitationId (Path Parameter):** The unique identifier of the invitation to be deleted. This parameter is required to ensure the correct invitation is targeted for deletion.