Standard Codes Gateway Service API

Table of Contents

Overview

The Standard Codes Gateway Service provides a robust API for retrieving standard code responses by their UUIDs. This service allows clients to search for multiple standard codes in a single request and receive consolidated responses.

The service is designed to handle both active and historical versions of content, supporting healthcare interoperability and Social Determinants of Health (SDOH) assessments.

UUID Types and Response Behavior

The system supports two different types of UUIDs that determine what content is returned:

1. Master Content UUID

- **Source**: Found as "id" at the root node
- **Behavior**: Returns the latest active version of the content
- **Use Case**: When you always want the most current version

2. Version-Specific UUID

- **Source**: Found as "identifier.value" in the response

- **Behavior**: Returns that exact version of the content
- **Use Case**: When you need a specific historical version

Content Status Indicators

Status	Indicator	Description
Active Content	`effectivePeriod.end` is null	Currently valid content
Inactive Content	`effectivePeriod.end` contains a date	Content deprecated on specified date

Content Types

Type	Resource Type	Description
SDOH Content	`resourceType="bundle"`	Social Determinants of Health content including assessments, goals, and interventions
List Content	`resourceType="valueSet"`	Standard code lists and terminology

Version Information

- `versionId` field indicates the specific version of the content
- Version tracking enables historical content retrieval and audit trails

---

API Endpoints

Search Responses by UUIDs

Retrieves standard code responses using a list of master UUIDs.

Endpoint: `POST /api/v1/standard-codes/responses/search`

Content Type: `application/json`

Authorization: No authentication currently required (permitAll)

Request and Response Structure

Request Body

```json
{
  "operation": "SEARCH",
  "parameters": {
    "uuids": [
      "89916142-7ea8-4cdd-b313-58131b297835",
      "ded9327f-8d0f-4148-a383-f291047831c4",
      "2ccc0965-f500-4602-bd91-83029b42b596",
      "4861b05d-c84e-44a0-aa29-0a7e981b84fd"
    ]
  },
  "priority": "HIGH"
}
```

Request Fields

Field	Type	Required	Description
operation	String	Yes	Operation type (currently only "SEARCH" is supported)
parameters.uuids	Array of UUIDs	Yes	List of master UUIDs to search for
priority	String	No	Priority of the request ("HIGH", "MEDIUM", "LOW")

Success Response Structure

```json
{
  "responseId": "550e8400-e29b-41d4-a716-446655440000",
  "requestId": "7a39f6f0-3db8-4fc3-b143-39aa13b53756",
  "timestamp": "2025-05-16T12:34:56.789Z",
  "status": "SUCCESS",
  "responses": [
    {
      // Standard code response data (JsonNode)
    },
    {
      // Standard code response data (JsonNode)
    }
  ],
  "error": null
}
```

Response Fields

Field	Type	Description
responseId	String	Unique ID for this response
requestId	String	Unique ID for the processed request
timestamp	ZonedDateTime	Timestamp when the response was generated
status	String	Status of the request ("SUCCESS", "ERROR")
responses	Array	List of standard code responses as JsonNode objects
error	Object	Error details (null if status is "SUCCESS")

Error Response Structure

```json
{
  "responseId": "550e8400-e29b-41d4-a716-446655440000",
  "requestId": "7a39f6f0-3db8-4fc3-b143-39aa13b53756",
  "timestamp": "2025-05-16T12:34:56.789Z",
  "status": "ERROR",
  "responses": null,
  "error": {
    "timestamp": "2025-05-16T12:34:56.789",
    "status": 400,
    "error": "Bad Request",
    "message": "Invalid request parameters",
    "path": "/api/v1/standard-codes/responses/search"
  }
}
```

Response Codes

Status Code	Description
200	Successfully retrieved responses
400	Invalid request (missing required fields, invalid UUIDs, etc.)
500	Internal server error

Sample Usage

Java Example

```java
import org.springframework.http.HttpEntity;
import org.springframework.http.HttpHeaders;
import org.springframework.http.MediaType;
import org.springframework.web.client.RestTemplate;

public JsonResponse searchStandardCodes(List<UUID> uuids) {
    RestTemplate restTemplate = new RestTemplate();

    HttpHeaders headers = new HttpHeaders();
    headers.setContentType(MediaType.APPLICATION_JSON);

    MasterUuidRequest request = new MasterUuidRequest();
    request.setOperation("SEARCH");

    UuidParameters parameters = new UuidParameters();
    parameters.setUuids(uuids);
    request.setParameters(parameters);

    request.setPriority("HIGH");

    HttpEntity<MasterUuidRequest> entity = new HttpEntity<>(request, headers);

    return restTemplate.postForObject(
        "http://localhost:8080/api/v1/standard-codes/responses/search",
        entity,
        JsonResponse.class
    );
}
```

cURL Example

bash

```bash
curl --location 'http://localhost:8080/api/v1/standard-codes/responses/search' \
--header 'Content-Type: application/json' \
--header 'Cookie: JSESSIONID=367DF526EF641BA5ED304CF9D422709E' \
--data '{
    "operation": "SEARCH",
    "parameters": {
      "uuids": [
        "89916142-7ea8-4cdd-b313-58131b297835",
        "ded9327f-8d0f-4148-a383-f291047831c4",
        "2ccc0965-f500-4602-bd91-83029b42b596",
        "4861b05d-c84e-44a0-aa29-0a7e981b84fd"
      ]
    },
    "priority": "HIGH"
  }'
```

Working with Response Data

Handling ValueSet Responses

When working with `resourceType="valueSet"` responses:

1. **Access Code Options**: Use the `answerOption` array to retrieve available code options
2. **Extract Code Information**: Each `valueCoding` contains:
   - `code`: The unique code identifier
   - `system`: The coding system (e.g., "SNOMEDCT")
   - `display`: Human-readable description
   - `extension`: Additional properties including unique IDs

Determining Content Status

java

```java
// Check if content is currently active
boolean isActive = response.getEffectivePeriod().getEnd() == null;
```

Getting Content Version

```java
// Retrieve the version identifier
String version = response.getMeta().getVersionId();
```

Key Response Fields

Field	Description
id	Master content UUID for fetching latest active version
identifier.value	Version-specific UUID for fetching exact version
resourceType	Content type ("valueSet" for lists, "bundle" for SDOH content)
status	Content status ("active", "draft", "retired", etc.)
meta.versionId	Version number of the content
effectivePeriod.start	When this version became effective
effectivePeriod.end	Deprecation date (null = active, date = deprecated)
answerOption	Available codes for valueSet resources
item	Code definitions and metadata container

Validation Rules

The API enforces the following validation rules:

- The `operation` field must not be blank
- The `parameters` field must not be null
- The `uuids` list must not be empty
- All UUIDs must be in valid UUID format (e.g., "550e8400-e29b-41d4-a716-446655440000")

Best Practices

1. Cache Responses

For frequently accessed standard codes, implement client-side caching to reduce server load and improve performance.

2. Error Handling

Implement robust error handling to manage different response codes and error conditions gracefully.

3. Bulk Requests

Group related UUIDs in a single request rather than making multiple individual requests to reduce network overhead.

4. Version Tracking

If specific versions are required, store the version-specific UUIDs rather than master content UUIDs to ensure consistency.

5. Priority Setting

Use the "priority" field appropriately - set to "HIGH" only for time-sensitive operations.

Implementation Considerations

Performance

- The service is optimized for batch retrieval
- Requesting multiple UUIDs in a single call is more efficient than multiple individual requests
- Response sizes can vary significantly depending on content complexity

Security

- Currently uses `permitAll()`, but production environments should implement appropriate authentication
- Consider using HTTPS to ensure secure data transmission
- Implement proper authorization controls for sensitive content

Monitoring

- Service includes logging for request tracking and debugging
- Each response includes unique `responseId` and `requestId` for request tracing
- Monitor response times and error rates for service health

Examples and Appendices

Content Type Comparison

The API supports two main content types with distinct characteristics:

ValueSet Resources

- **Resource Type**: `resourceType="valueSet"`
- **Structure**: Collection of code values with descriptions
- **Organization**: Flat list of options
- **Usage**: Reference data and terminology
- **Examples**: Care Team Member Roles, Tribal Affiliations

Bundle Resources

- **Resource Type**: `resourceType="bundle"`
- **Structure**: Hierarchical content with structured items
- **Organization**: Assessment questions, goals, and interventions
- **Usage**: Complex SDOH (Social Determinants of Health) content
- **Examples**: Food Insecurity Assessment, Digital Access Assessment

SDOH Questionnaire Structure

Social Determinants of Health assessments follow a standardized structure:

1. **Assessment Questions Section**: Standardized questions for evaluation
2. **Goals Section**: Available options for patient goals
3. **Interventions Section**: Available procedures and resources

This standardized approach enables healthcare providers to consistently evaluate social factors impacting health outcomes and connect patients with appropriate resources.

Tribal Affiliation ValueSet

The Tribal Affiliation ValueSet provides standardized codes for documenting tribal affiliations according to USCDI v3 requirements:

- **Purpose**: Consistent documentation of tribal affiliations in healthcare records
- **System**: Uses `http://terminology.hl7.org/CodeSystem/v3-TribalEntityUS`
- **Content**: Comprehensive list of federally recognized US tribes
- **Integration**: Compatible with other USCDI v3 data elements for interoperability
- **Usage**: Clinical documentation, demographics, and reporting systems

---

API Versioning

This API is versioned in the URL path (`/api/v1/`). Future versions will be released under different version paths (e.g., `/api/v2/`) to maintain backward compatibility.

Notes

- Response data format depends on the retrieved standard codes
- Responses are returned in the same order as requested UUIDs
- If a UUID is not found, the corresponding response entry will be null
- Empty `effectivePeriod.end` indicates active content
- Populated `effectivePeriod.end` indicates content deprecation date

Support

For any issues or questions regarding this API, please contact the CMT team.

Document Version: 1.0
Last Updated: May 29, 2025
API Version: v1