CSP Server Configuration API

Version 1.0.0.3

Paths

`/v1/updates/{cspId}`

GET /v1/updates/{cspId}

Description

Retrieves a list of available updates, for registered modules of the CSP. See the "register" API call on how to register modules

Parameters

Name	Located in	Description	Required	Schema
cspId	path	a unique identifier that defines a Registered and Known CSP. The csp identifier follows the UUID formatted as text, for 36 characters total, arranged as 8-4-4-4-12. Example = 123e4567-e89b-12d3-a456-426655440000	Yes	⇄string

Responses

Code	Description	Schema
200	List of available updates by module	⇄ ▼ **UpdateInformation {** *contains update information for the selected CSP* dateChanged: ► string * available: ► { } }
404	cspId not found; failure to identify cspId presented	⇄ ▼ **ResponseError {** *Failed Response object* responseCode: ► integer * responseText: ► string * responseException: string * }

500 Systemic failure	⇄	▼ **ResponseError {** *Failed Response object* responseCode: ▶ integer * responseText: ▶ string * responseException: string * }

Try this operation

/v1/heartbeat/{cspId}

POST /v1/heartbeat/{cspId}

Description

Submits a heartbeat with a body that contains information of the CSP

Parameters

Name	Located in	Description	Required	Schema
cspId	path	a unique identifier that defines a Registered and Known CSP. The csp identifier follows the UUID formatted as text, for 36 characters total, arranged as 8-4-4-4-12. Example = 123e4567-e89b-12d3-a456-426655440000	Yes	⇄ string
heartbeatInfo	body	the heartbeat information	Yes	⇄ ▼ **HeartBeatInfo {** *Heartbeat object* name: ▶ string * recordDateTime: ▶ string * moduleInfo: ▶ **ModulesInfo { }** }

Responses

Code	Description	Schema

200	Successful response	⇄	▼ **Response {** *Accepted Response object* responseCode: ▶ integer * responseText: ▶ string * }
404	cspId not found; failure to identify cspId presented	⇄	▼ **ResponseError {** *Failed Response object* responseCode: ▶ integer * responseText: ▶ string * responseException: string * }
500	Systemic failure	⇄	▼ **ResponseError {** *Failed Response object* responseCode: ▶ integer * responseText: ▶ string * responseException: string * }

Try this operation

/v1/register/{cspId}

POST /v1/register/{cspId}

Description

Register a NEW csp or register for an existing CSP the modules that are being installed

Parameters

Name	Located in	Description	Required	Schema
cspId	path	a unique identifier that defines a Registered and Known CSP. The csp identifier follows the UUID formatted as text, for 36 characters total,	Yes	⇄ string

	arranged as 8-4-4-4-12. Example = 123e4567-e89b-12d3-a456-426655440000			
cspRegistration	body	a block of information to register the CSP being installed	Yes	⇄ ▼ **Registration {** *Registration object for a new CSP* name: ► string * registrationDate: ► string * externalIPs: ►[] internalIPs: ►[] registrationIsUpdate: ► boolean * contacts: ►[] moduleInfo: ► **ModulesInfo { }** }

Responses

Code	Description	Schema
200	received when the CSP is registered successfully	⇄ ▼ **Response {** *Accepted Response object* responseCode: ► integer * responseText: ► string * }
400	received when CSP updates entries with "registrationIsUpdate = false"	⇄ ▼ **ResponseError {** *Failed Response object* responseCode: ► integer * responseText: ► string * responseException: string * }
		▼ **ResponseError {** *Failed Response object* responseCode: ►

		responseCode:	integer *
404	received when CSP requests update but original entry does not exist OR when cspId is not recognized	⇄ responseText:	► string *
		responseException:	string *
		}	
		▼ **ResponseError {** *Failed Response object*	
		responseCode:	► integer *
500	received when the CSP is not registered due to system error	⇄ responseText:	► string *
		responseException:	string *
		}	

Try this operation

/apiversion/{cspId}/{apiVersion}

GET /apiversion/{cspId}/{apiVersion}

Description

A simple API call to allow a client to use a particular version of the API. In the future, multiple API versions may be available but backwards compatibility should be ensured.

Parameters

Name	Located in	Description	Required	Schema
cspId	path	a unique identifier that defines a Registered and Known CSP. The csp identifier follows the UUID formatted as text, for 36 characters total, arranged as 8-4-4-4-12. Example = 123e4567-e89b-12d3-a456-426655440000	Yes	⇄ string

apiVersion	path	the api version supported, without dots, eg. 1.0.0 -> 100	Yes	⇄integer

Responses

Code	Description	Schema
200	cspId is identified and versions supported are reported in payload	⇄ ▼**ApiVersions {** *object describing versions supported by API* reportedVersion: ►**VersionDefinition** integer * reportedVersionString: ► string * supportedVersions: ►**VersionsList[]** }
400	version in query is not available	⇄ ▼**ResponseError {** *Failed Response object* responseCode: ► integer * responseText: ► string * responseException: string * }
404	cspId not found; failure to identify cspId presented	⇄ ▼**ResponseError {** *Failed Response object* responseCode: ► integer * responseText: ► string * responseException: string * }
500	Systemic failure	⇄ ▼**ResponseError {** *Failed Response object* responseCode: ► integer * responseText: ► string * responseException: string * }

Try this operation

Models

ApiVersions

ApiVersions {

object describing versions supported by API

reportedVersion: **VersionDefinition** integer *

reportedVersionString: string *

supportedVersions: **VersionsList[]**

}

VersionDefinition

VersionDefinition integer * minimum:10000 maximum:90000

the version definition used for communicating with the CSP APIs; it has a visual and a numeric format. The visual format can be created by splitting the numeric at specific points, (0 based) 1, 2 example: 12000 split as 1.2.000

VersionsList

VersionsList[

Version { }

]

Version

Version {

version: **VersionDefinition** integer *

versionString: string

path: string

dateReleased: string

}

Registration

Registration {

Registration object for a new CSP

name: string *

registrationDate: string *

externalIPs: **[]**

```
    internalIPs:           ▶[]
    registrationIsUpdate: ▶ boolean *
    contacts:              ▶[]
    moduleInfo:            ▶ModulesInfo { }
  }
```

ContactDetails

```
  ▼ContactDetails {
    personName:  string *
    personEmail: ▶ string *
    contactType: ▶ string *
  }
```

HeartBeatInfo

```
  ▼HeartBeatInfo {
    Heartbeat object
    name:           ▶ string *
    recordDateTime: ▶ string *
    moduleInfo:     ▶ModulesInfo { }
  }
```

Response

```
  ▼Response {
    Accepted Response object
    responseCode: ▶ integer *
    responseText: ▶ string *
  }
```

ResponseError

```
  ▼ResponseError {
    Failed Response object
    responseCode:      ▶ integer *
```

responseText: ► string *

responseException: string *

}

ModulesInfo

▼ **ModulesInfo {**

The json object that contains information on the modules installed

⇄ modules: ► **[]**

data: ► **{ }**

}

ModuleData

▼ **ModuleData {**

Map of data specific to a module

fullName: string *

⇄ version: ► **VersionDefinition** integer *

installedOn: ► string *

active: boolean *

hash: string *

}

UpdateInformation

▼ **UpdateInformation {**

contains update information for the selected CSP

⇄ dateChanged: ► string *

available: ► **{ }**

}

ModuleUpdateInfo

▼ **ModuleUpdateInfo {**

list of available updates for a module

name: string *

description: string *

```
⇄   version:       ▶VersionDefinition integer *
    released:      ▶ string *
    isIncremental: ▶ boolean *
    hash:          ▶ string *
}
```