chapter 33

REST preload requests

When an InsuranceSuite server is first started, initial REST API requests may be slow to execute. Guidewire allows you to define "preload" requests for REST APIs to ensure that initial API requests from external callers are processed quickly.

Preload requests are REST API calls that are executed automatically after server startup to warm up endpoints. These calls are treated similarly to regular API calls, triggering loading of configuration files, just-in-time compilation, cache loading, and so on. However, they are made by the InsuranceSuite application itself and not by an external caller.

This mechanism is primarily intended for use with Cloud API endpoints, but it can also be used with REST endpoints built directly on the InsuranceSuite REST Framework.

This mechanism complements the REST servlet preload option, which is controlled by the `PreloadRestServletConfig` configuration parameter. This configuration parameter must be set to `true` for preload requests to be executed.

For more on warming up endpoints, see the *Application Guide*.

Defining preload requests

Preload requests can be defined by adding JSON files in the `configuration/config/integration/preload` directory or any of its subdirectories. You must create the `preload` directory in the correct place for the application to execute the calls.

Each JSON file holds a single preload request object. Preload request objects define a single API call. Preload requests are executed in alphabetical order based on the name of the JSON file.

In each preload request object, there are two required properties:

Field	**Data type**	**Description**
`method`	String	The HTTP method for the subrequest, such as GET or POST. The string is not case sensitive.
`path`	String	The path for the request, relative to the servlet root (for example, `/admin/v1/users`).

For example, the following JSON file defines a minimal preload request for GETing the claim graph schema:

```
{
  "path": "/claim/v1/graph-schema",
  "method": "get"
}
```

There are also optional properties that define call behavior, such as body, description, and headers. The full schema for preload request objects is found at `configuration/config/integration/schemas/gw/core/pl/preload/preload_pl-1.0.schema.json`.

Note: The `PreloadRestServletConfig` parameter must be set to `true` for preload requests to execute.

Preload requests can make use of the Composite API to execute a sequence of related requests. This is useful to ensure that all endpoints for a specific workflow have been warmed up. For example, the following preload request performs all actions required to create and quote a personal auto policy in PolicyCenter, starting with creating an account.

```
{
  "path":"/composite/v1/composite",
  "method":"POST",
  "body":{
    "requests":[
      {
        "method":"post",
        "uri":"/account/v1/accounts",
        "body":{
          "data":{
            "attributes":{
              "initialAccountHolder":{
                "contactSubtype":"Person",
                "firstName":"Tamsin",
                "lastName":"Tester",
                "primaryAddress":{
                  "addressLine1":"2850 S. Delaware St.",
                  "city":"San Mateo",
                  "postalCode":"94403",
                  "state":{
                    "code":"CA"
                  }
                }
              },
              "initialPrimaryLocation":{
                "addressLine1":"2850 S. Delaware St.",
                "city":"San Mateo",
                "postalCode":"94403",
                "state":{
                  "code":"CA"
                }
              },
              "producerCodes":[
                {
                  "id":"pc:16"
                }
              ],
              "organizationType":{
                "code":"other"
              }
            }
          }
        },
        "vars":[
          {
            "name":"accountId",
            "path":"$.data.attributes.id"
          },
          {
            "name":"driverId",
            "path":"$.data.attributes.accountHolder.id"
          }
        ]
      },
      {
        "method":"post",
        "uri":"/job/v1/submissions",
        "body":{
          "data":{
            "attributes":{
              "account":{
                "id":"${accountId}"
              },
              "baseState":{
                "code":"CA"
              },
              "jobEffectiveDate":"2022-08-01",
              "producerCode":{
                "id":"pc:16"
              },
              "product":{
                "id":"PersonalAuto"
              }
            }
          }
        },
        "vars":[
          {
            "name":"jobId",
            "path":"$.data.attributes.id"
          }
        ]
      },
      {
        "method":"patch",
        "uri":"/job/v1/jobs/${jobId}/questions",
        "body":{
          "data":{
            "attributes":{
              "answers":{
                "PACurrentlyInsured":{
                  "choiceValue":{
                    "code":"newdriver"
                  }
                }
              }
            }
          }
        }
      },
      {
        "method":"post",
        "uri":"/job/v1/jobs/${jobId}/lines/PersonalAutoLine/coverages",
        "body":{
          "data":{
            "attributes":{
              "pattern":{
                "id":"PALossOfUseCov"
              }
            }
          }
        }
      },
      {
        "method":"post",
        "uri":"/job/v1/jobs/${jobId}/lines/PersonalAutoLine/vehicles",
        "body":{
          "data":{
            "attributes":{
              "make":"Toyota",
              "model":"Camry",
              "modelYear":2010,
              "costNew":{
                "amount":"33000",
                "currency":"usd"
              },
              "licenseState":{
                "code":"CA"
              },
              "vin":"14HEW8RLGMDSP03AA"
            }
          }
        },
        "vars":[
          {
            "name":"vehicleId",
            "path":"$.data.attributes.id"
          }
        ]
      },
      {
        "method":"post",
        "uri":"/job/v1/jobs/${jobId}/lines/PersonalAutoLine/vehicles/${vehicleId}/coverages",
        "body":{
          "data":{
            "attributes":{
              "pattern":{
                "id":"PARentalCov"
              },
              "terms":{
                "PARental":{
                  "choiceValue":{
                    "code":"60/20"
                  }
                }
              }
            }
          }
        }
      },
      {
        "method":"patch",
        "uri":"/job/v1/jobs/${jobId}/contacts/${driverId}",
        "body":{
          "data":{
            "attributes":{
              "dateOfBirth":"1980-10-10",
              "licenseNumber":"CA7732839",
              "licenseState":{
                "code":"CA"
              },
              "numberOfAccidents":{
                "code":"0"
              },
              "numberOfViolations":{
                "code":"0"
              },
              "policyNumberOfAccidents":{
                "code":"0"
              },
              "policyNumberOfViolations":{
                "code":"0"
              }
            }
          }
        }
      },
      {
        "method":"post",
        "uri":"/job/v1/jobs/${jobId}/lines/PersonalAutoLine/vehicles/${vehicleId}/drivers",
        "body":{
          "data":{
            "attributes":{
              "percentageDriven":100,
              "policyDriver":{
                "id":"${driverId}"
              }
            }
          }
        }
      },
      {
        "method":"patch",
        "uri":"/job/v1/jobs/${jobId}/lines/PersonalAutoLine/vehicles/${vehicleId}/modifiers/PAAntiLockBrakes",
        "body":{
          "data":{
            "attributes":{
              "booleanModifier":true
            }
          }
        }
      },
      {
        "method":"post",
        "uri":"/job/v1/jobs/${jobId}/quote"
      }
    ]
  }
}
```

Behavior of preload requests

Preload requests are processed similarly to requests from caller applications.

However, data in preload POST and PATCH requests is not committed to the database. Cloud API preload requests include the `GW-DoNotCommit` header by default, which ensures that the data is not committed. For custom REST APIs, a custom header needs to be included to prevent the data from being committed.

Note that you might need to implement additional logic to prevent preload requests from triggering downstream systems, if necessary.

Authentication for preload requests

If no authentication information is provided in the preload request object, preload requests are executed using the credentials of the unrestricted user `su`. Optionally, you can use the `asUser` field in the preload request object to specify the id of another user as the caller. Use the public id of the user, which is returned in the `id` field when retrieving a user through Cloud API.

Authentication plugins `RestAuthenticationSourceCreatorPlugin` and `RestAuthenticationServicePlugin` handle authentication specially for preload requests. These plugins must be in use for preload requests to be properly executed.

Logging preload requests

Preload requests do not trigger normal logging and observability calls. Instead, they are logged only with INFO messages to the console if the request is completed successfully:

```
INFO Preload request '<name of preload request>’ completed successfully
```

If a request fails, an error message is logged to the console with the HTTP status code.