CID Info

Wednesday, November 22, 2017 1:46 PM

CID QA Testing Jenkins Project Setup

qa-cid-test Git repo contains TestRunner.groovy file that CID calls. CID refers to the Continuing Integration and Delivery system by Developer Services.

The workflow looks like this:
CID:
`CID –[calls]-> TestRunner.groovy –[calls]-> protractor with qa-test-us`

QA ("unit test"):
`qa-cid-test Jenkinsfile –[calls]-> TestRunner.groovy –[calls]-> protractor cid.conf.js`

IMPORTANT: The Jenkinsfile in qa-cid-test project is only for QA's own test driving of TestRunner.groovy. **CID only calls TestRunner.groovy**, not Jenkinsfile, from qa-cid-test Git repo.

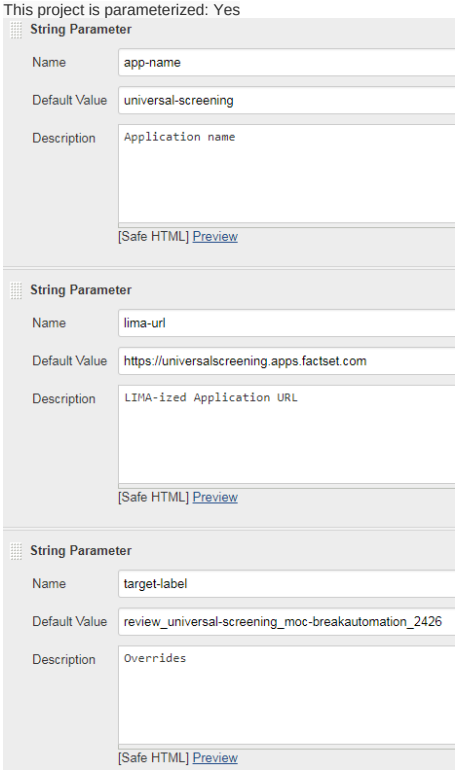
Git repo: https://gitlab.factset.com/app-qa-automation/qa-cid-test
Jenkins QA CID project: qa-cid-test

> Note: Engineering repo has a `factset.json` that contains parameters for CID.

qa-cid-test Jenkins Project Configuration

Office 365 Connector

This project is parameterized: Yes



How it works

The qa-cid-test repo has TestRunner.groovy script. CID will call `RunTestsForApp(cidData)` function in that script. And CID passes a parameters to it, in a form of a Groovy `map` object. We use the following entries in that map object:

app-name
Application name
Ex: universal-screening

lima-url
The URL of Web application, e.g.
Ex: https://universalscreening.apps.factset.com

target-label
FDSAv2 overrides
Ex: review_universal-screening_moc-breakautomation_2426

QA's own testing

QA's own test of the TestRunner.groovy script is driven by Jenkinsfile in qa-cid-test GitHub repo. Go to qa-cid-test project in Jenkins and click **Build with Parameters** to execute (non-CID) test run.

Jenkinsfile

That Jenkins file is a Jenkins Pipeline script, which is in Groovy language.
Structure:

```
node('cid') {
```

```
    // Wipe the workspace so we are building completely clean
    deleteDir()

    stage('Initialize') {
        echo 'Initializing...'
    }

    stage('Checkout') {
        echo 'Getting source code...'
        checkout scm
    }

    stage('Build') {
        echo 'Installing dependencies...'
    }

    stage('Test') {
        echo 'Testing...'
        // <<Your test here>>
  }
}
```

The 'Test' stage does this:

- Load TestRunner.groovy script
- Call RunTestsForApp(param) function in it, passing parameter to it.
  The parameters are similar to that sent by CID.

  Note: Currently the app-name and lima-url in Jenkins file are hard-coded to run Universal Screening test.

Configuration

At the top of TestRunner.groovy, there is the main configuration. It is in the form of Groovy `map` data structure.

```
@Field def config = [
  "serialNumberAPIBaseURL": "http://tellusstgb01:8080",
  "stage": "edge",
  "vmsUserName": "FDSQAR_C",
  "browserVersion": "61",
  "protractorVersion": "^5.1.2",
  "seleniumGridHub": "http://tellusdevb03.pc.factset.com:4444/wd/hub",
  "applicationMapping": [
    "universal-screening": [
      "package": "qa-test-us",
      "projectType": "protractor",
      "fdsav2Overrides": "us-web-release",
      "baseURLSuffix": "/#/?QA_MODE=1"
    ]
  ]
]
```

Note: We currently hard-code the configuration for reliability reasons: if the configuration were a JSON file in external Web server, that Web server would probably need to have high availability.

You will need to add each application under applicationMapping, and use the app-name as the key, for example, `"universal-screening"`.

"package": Artifactory package name, e.g. "qa-test-us",
"projectType": "protractor" or "ruby" (Selenium)
"fdsav2Overrides": FDSAv2 override, e.g. "us-web-release"
"baseURLSuffix": Appended to application URL, e.g. "/#/?QA_MODE=1"

test-config.json

This project also dynamically creates test-config.json in workspace. The test script will read it for overrides, etc.

```
    def json = """
    {
        "Stage": "${config.stage}",
        "FDSAv2Stage": "${config.stage}",
        "FDSAv2Overrides": "${config.fdsav2Overrides}",
        "HTMLReports": "Staging",
        "PrebuildName": "",
        "SGBrowserName": "chrome",
        "SGBrowserVersion": "${config.browserVersion}",
        "VMSUserID": "FDSQAR_C",
        "SerialNumber": "${config.serialNumber}",
        "UserName": "automation.tc${config.serialNumber}",
        "Password": "TellusAut0mation",
        "TestEnvironment": "SGE",
        "SeleniumGridHub": "${config.seleniumGridHub}"
    }
    """
```

Examples

- Example TestRunner.groovy
- Example Jenkinsfile

Jenkins Pipeline Scripting Tips

Jenkins Pipeline CID scripting tips

Scripting Team's Responsibility for CID

The QA Automation test scripting team is responsibility for the following tasks for CID:

1. Supply **cid.conf.js:** including `specs`, which specifies a set of tests to run. Protractor will execute that file, *as is*.
2. Produce a test package of the version corresponding to the current DEVEL stage of Online Tracker in Artifactory

Reference

- https://www.cyotek.com/blog/using-parameters-with-jenkins-pipeline-builds

Example Jenkinsfile

Thursday, December 14, 2017 11:18 AM

Note: See the latest changes in the repo.

```
/**
 * QA Automation test runner.
 */

node('cid') {
    // Wipe the workspace so we are building completely clean
    deleteDir()

    stage('Initialize') {
        echo 'Initializing...'
    }

    stage('Checkout') {
        echo 'Getting source code...'
        checkout scm
    }

    stage('Build') {
        echo 'Installing dependencies...'
    }

    stage('Test') {
        echo 'Testing...'

        def cidData = [
            "app-name"        : "cid-test-service",
            "repo-url"        : "git@gitlab.factset.com:bbatha/cid-test-service.git",
            "commit"          : "8428dccb55db6991687a755bb6d5e5bcef90d39d",
            "factset-json"    : [
              "name": "cid-test-service",
              "end_to_end_tests": [
                "lima_staging_url": "https://universalscreening.staging-cauth.factset.com",
                "lima_inhouse_url": "https://universalscreening.inhouse-cauth.factset.com",
                "lima_prod_url": "https://universalscreening.apps.factset.com"
              ],
              "owners": [
                "page": "cid_oncall",
                "admin_email": "cid-admin@factset.com"
              ],
              "type": "service",
              "scripts": [
                "test": "npm test"
              ],
              "config_version": 0.2,
              "deployment": [
              "factsetio": [
                "buildpacks": [
                  [
                    "url": "https://github.com/ryandotsmith/null-buildpack"
                  ],
                  [
                    "url": "http://artifactory.factset.com/artifactory/repo/legacy-node-platform-buildpack/legacy-node-platform-buildpack.tgz"
```

```
                ]
              ],
              "formation": [
                "web": [
                  "quantity": 2,
                  "size": "1X"
                ]
              ]
            ]
          ]
        ],
        "fdsav2-url"      : "cid-test-service.services.nprod.factset.com",
        "lima-url": "https://universalscreening.apps.factset.com",
        "e2e-label"       : "e2e_cid-test-service_4f8fce64-02ad-4eef-bba5-81fb4782f90b",
        "merge-request-id": "83357",
        "repo-id"         : "bbatha/cid-test-service"
      ]

      // CONFIG
      cidData["app-name"] = "universal-screening"
      cidData["lima-url"] = "https://universalscreening.apps.factset.com"

      def testRunner = load "TestRunner.groovy"

      testRunner.RunTestsForApp(cidData)

  }

  /*
  stage('Publish') {
      echo 'Publishing Test Coverage...'
      publishHTML (target: [
              allowMissing: false,
              alwaysLinkToLastBuild: false,
              keepAll: true,
              reportDir: 'coverage/lcov-report',
              reportFiles: 'index.html',
              reportName: "Application Test Coverage"
      ])
  }
  */

}
```

Example TestRunner.groovy

Thursday, December 14, 2017 11:18 AM

Note: See the latest changes in the repo.

```
import groovy.json.JsonSlurper
import groovy.json.JsonSlurperClassic
import groovy.transform.Field

////////////////////////////////////////////////////////////////////////////////
// Configuration
// @Field def configUrl = "http://tellusb01.pc.factset.com/cid/qa-cid-test.config.json"
@Field def config = [
  "serialNumberAPIBaseURL": "http://tellusstgb01:8080",
  "stage": "edge",
  "vmsUserName": "FDSQAR_C",
  "browserVersion": "61",
  "protractorVersion": "^5.1.2",
  "seleniumGridHub": "http://tellusdevb03.pc.factset.com:4444/wd/hub",
  "applicationMapping": [
    "universal-screening": [
      "package": "qa-test-us",
      "projectType": "protractor",
      "fdsav2Overrides": "us-web-release",
      "baseURLSuffix": "/#/?QA_MODE=1"
    ]
  ]
]

////////////////////////////////////////////////////////////////////////////////

def loadConfig(url) {
    def response = sh(script: "curl -s ${url}", returnStdout: true).trim()
    return (new JsonSlurperClassic()).parseText(response)
}

/**
 * Allocate serial number and set override.
 *
 * @param serialNumberAPIBaseURL e.g. http://tellusstgb01:8080
 * @param fdsav2Overrides FDSAv2 overrides
 * @param stage "edge" by default
 * @param vmsUserName "FDSQAR_C" by default
 * @return
 */
def acquireSerial(serialNumberAPIBaseURL, fdsav2Overrides, stage="edge", vmsUserName="FDSQAR_C") {
    def reqStr = """{\"RequestJson\": {\"serialnumber\": \"random\", \"vmsusername\": \"${vmsUserName}\",
\"fdsav2stage\": \"${stage}\", \"fdsav2overrides\": \"${fdsav2Overrides}\"}}"""
    def response = sh(script: "curl -sk -XPOST -d '$reqStr' -H \"Content-Type: application/json\" -H \"Accept:
application/json\" ${serialNumberAPIBaseURL}/tellusci/ci/request",
            returnStdout: true)
    echo "acquireSerial curl response: $response"
    def jsonParser = new JsonSlurper()
    def json = jsonParser.parseText(response)
    return json.ResponseJson.serialnumber
}

def releaseSerial(serialNumberAPIBaseURL, serialnumber) {
    def reqStr = "{\"RequestJson\":{\"serialnumber\":$serialnumber}}"
    def response = sh(script: "curl -sk -XPOST -d $reqStr -H \"Content-Type: application/json\" -H \"Accept:
application/json\" ${serialNumberAPIBaseURL}/tellusci/ci/release",
            returnStdout: true)
    echo "release curl response: $response"
    def jsonParser = new JsonSlurper()
    def json = jsonParser.parseText(response)
    return json.ResponseJson.serialnumber
```

```
}

/**
 * Get version for stage from Tracker API.
 * @param stage default "devel"
 * @param debug true for debug output
 * @return
 */
def getOnlineVersion(stage = "devel") {
    /*
      Example response JSON:
      {"actions":[{"label_name":"online_2017_07_11_001","cluster":"fxdev1-
a","stage":"devel","label_minor":5,"status":1,"message":null,"stamp":"2017-07-11
18:41:10-04","user":null,"label_id":2736,"platform":"Fonix","id":141857,"type":"promote","patch":2,"label_versio
n":203,"overwritten":0}],"last_stamp":"2017-07-11 18:41:10-04","status":1,"last_status":1}
     */
    def trackerUrl = "http://tracker.factset.com/build_status?stage=${stage}&type=promote"
    def response = trackerUrl.toURL().text
    echo(response)
    def jsonSlurper = new JsonSlurper()
    def json = jsonSlurper.parseText(response)
    def actions = json.actions
    def rv = null
    if (actions.size() > 0) {
        def item = actions[0]
        rv = item.label_version
    }
    return rv
}

def RunProtractor(config) {
    // Initialize environment
    def nodeHome = tool name: 'node 6.x.x LTS', type: 'jenkins.plugins.nodejs.tools.NodeJSInstallation'
    env.PATH = "${nodeHome}/bin:${env.PATH}"

    sh """
    node --version
    npm --version
    """

    def json = """
    {
        "Stage": "${config.stage}",
        "FDSAv2Stage": "${config.stage}",
        "FDSAv2Overrides": "${config.fdsav2Overrides}",
        "HTMLReports": "Staging",
        "PrebuildName": "",
        "SGBrowserName": "chrome",
        "SGBrowserVersion": "${config.browserVersion}",
        "VMSUserID": "FDSQAR_C",
        "SerialNumber": "${config.serialNumber}",
        "UserName": "automation.tc${config.serialNumber}",
        "Password": "TellusAut0mation",
        "TestEnvironment": "SGE",
        "SeleniumGridHub": "${config.seleniumGridHub}"
    }
    """

    writeFile file: "test-config.json", text: json
    writeFile file: "SerialNumber.txt", text: "$config.serialNumber"

    def protractorSuffix = config.protractorVersion ? "@${config.protractorVersion}" : ""
    def versionSuffix = config.version ? "@^${config.version}" : ""

    sh """
        npm prune
        npm install protractor${protractorSuffix}
        npm install @fds/${config.package}${versionSuffix}
        ./node_modules/.bin/protractor --params.path \$(pwd) --baseUrl ${config.baseURL} node_modules/@fds/
```

```
${config.package}/cid.conf.js
    """
}

def RunTestsForApp(cidData) {
    // app-name string Name of the app
    // repo-url string URL to the git repo
    // commit string The commit being built
    // factset-json map The factset.json of the app
    // fdsav2-url string The FDSAv2 URL for the app. i.e; <appname>.nprod.services.factset.com
    // e2e-label string The label generated for holding the deployment to be tested

    echo "RunTestForApp with CID data:"
    echo cidData.toString()

    // def config = loadConfig(configUrl)
    echo "Using Tellus configuration:"
    echo config.toString()

    def appInfo = config.applicationMapping[cidData['app-name']]
    echo "projectType: ${appInfo.projectType}"

    def trackerVersion = getOnlineVersion()
    echo "Tracker version: ${trackerVersion}"

    // Store configuration
    config.version = trackerVersion ? "${trackerVersion}.0.0" : ""
    config.baseURL = cidData["lima-url"] + appInfo.baseURLSuffix ?: ""
    config.package = appInfo.package
    config.fdsav2Overrides = appInfo.fdsav2Overrides

    def serialNumber = null

    try {
        config.serialNumber = acquireSerial(config.serialNumberAPIBaseURL,
                appInfo.fdsav2Overrides,
                config.stage,
                config.vmsUserName)
        echo "Acquired serial number: $config.serialNumber"

        switch (appInfo.projectType) {
            case "selenium":
//            RunSelenium(map)
                break
            case "protractor":
                RunProtractor(config)
                break
            default:
                RunProtractor(config)
                break
        }

    } finally {
        if (config.serialNumber) {
            releaseSerial(config.serialNumberAPIBaseURL, config.serialNumber)
        }
    }

    echo "Test completed"
}

return this
```

CID Jenkins Pipeline tips

Thursday, December 14, 2017 11:23 AM

Although Jenkins Pipeline script is Groovy language, there are some environment differences and restrictions.

Environment Differences:

- It must have node('cid') as top level for CID environment.
- Use `sh` instead of `process` to execute command line commands (the latter is not permitted)
- Use `echo` instead of `println` (the latter is not available). `echo` messages will show up in CID console log.
- Use `sh` with curl for calling Web service, http library call is not allowed in Jenkins

Issues or differences:

1. No source level debugger (cannot step through code and inspect data)
2. global function does not work with node(...)
3. In pipeline script, switch statement with return from case does not work (but this workaround works: assignment to variable within switch case, then return from outside switch). But regular groovy works.
4. Script name becomes "Script1.groovy" in exception's stack trace, different from original name
5. Exception stack trace does not correspond to the lines of your pipeline source
6. JsonSlurper is huge pain with exception (it's async):
   java.io.NotSerializableException: groovy.json.internal.LazyMap
   => Use JsonSlurper**Classic** instead (synchronous)

Restrictions:

1. HashMap is forbidden in Jenkins, by default with error: "Scripts not permitted to use new java.util.LinkedHashMap". Solution needs plugin configuration change to allow it.
2. enum is forbidden by default with error: "Scripts not permitted to use new java.util.LinkedHashMap"
   Workaround: use class and static constants, for example:
   ```
   class ProjectTypes {
       public final static int Protractor = 1
       public final static int Selenium = 2
   }
   ```
3. Scripts not permitted to use method java.lang.Class isInstance (instanceof)