Continuous Delivery as Code with Jenkins Workflow

Jesse Glick, CloudBees (@tyvole)

Jenkins Workflow as Code

Why Workflow?

- Complex build/test/deploy pipelines
  - traditional Jenkins projects too rigid
  - sometimes need custom logic
- Stages, human input, parallelism, …
- Long-running build steps survive Jenkins restarts

Why “as code”?

- Overall job definition is a script
  - calls your build tools and scripts for details
- Script can be versioned alongside project sources
  - experimental branches
  - code review!
- Keep less configuration in `$JENKINS_HOME`

Docker

Workflow plugin for Docker

- Simple entry point to Docker-related functions
- Build images from `Dockerfile`
- Test images in temporary containers
- Push to Docker Hub or private Docker registry

Docker as a build environment

- Run build steps inside a container!
  - use a standard image from Docker Hub
  - or from your company's registry
  - or build it from project sources
- Use generic Linux slaves
- Reproduce CI environment on your laptop
- Empowers developers to use the right tools

```
docker.image('maven:3.3.3-jdk8').inside {sh 'mvn verify'}
```

Custom DSLs

Defining new variables

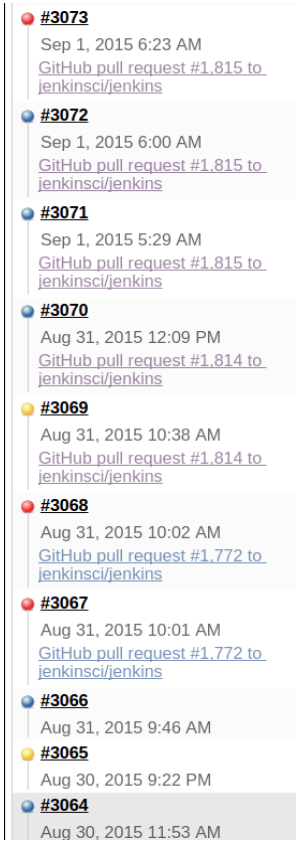
- `workflowLibs.git` holds sitewide definitions
- Had a `src` dir to define classes
- Now supports a `vars` subdirectory
- Scripts here can act like global variables
  - create a new DSL entry point, like `docker`

Defining new functions

- Vars act as functions, if you define a `call` method
  - like defining a new step, but in Groovy!
- Groovy builder pattern also possible
  - use: `myThing {prop = 'value'}`

Building multiple branches

Why can’t Jenkins branch?



- Can use branch wildcards in Git (etc.)
  - mixes up changelog, status, test results
  - no way to adjust build steps per branch
- Can create one project per branch
  - painful to maintain without extra tools
- Most people need this!

Multibranch workflows

- Branch with a `Jenkinsfile` → one subproject
  - that is your Workflow script
  - just `checkout scm` to get full source tree
- Can edit `Jenkinsfile` in your branch
  - revision matches sources

Branch sources

- Generic repository inspection
  - Git
  - Subversion
  - Mercurial
- Dedicated GitHub support
  - faster inspection using GitHub API
  - webhooks for instant notification

CJP extra branch sources

- Extended GitHub support
  - pull requests
  - commit status notification
- BitBucket
  - similar to GitHub
- Proposed integrations
  - Stash
  - Gerrit
  - validated merge

Organization folders

Organization folders

- Before: custom scripting just to add all 100 repos
- New folder type: “organization”
  - each item is a multibranch Workflow project
  - adds/removes projects automatically
- Only configuration is org name + credentials
  - one step closer to “code as config”

Organization types

- GitHub (organization or user)
  - private repositories supported
  - GitHub Enterprise
- CJP: BitBucket, Stash proposed

What to expect

Available now

- Released to update center
  - Git, Subversion, Mercurial integrations
  - Workflow custom DSLs (1.10)
  - CloudBees Docker Workflow
  - *Multi-Branch Project (freestyle)*
- Experimental update center
  - Workflow: Multibranch
  - *Literate, YAML Project (declarative)*

Coming soon

- To experimental update center
  - Organization folders incl. Workflow provider
  - GitHub integration
- To CJP experimental update center
  - GitHub pull requests
  - BitBucket integration

Future directions

- Cleanup before 1.0: UI, APIs, minor features
- GitHub: one-step OAuth & webhook setup
- Autobuild any `pom.xml`, `Rakefile`, …
- Specify job properties in `Jenkinsfile`
  - parameters, concurrency, …
- Restricted build scripts for pull requests