Syntax Comparison

When Jenkins Pipeline was first created, Groovy was selected as the foundation. Jenkins has long shipped with an embedded Groovy engine to provide advanced scripting capabilities for admins and users alike. Additionally, the implementors of Jenkins Pipeline found Groovy to be a solid foundation upon which to build what is now referred to as the "Scripted Pipeline" DSL. [1: Domain-Specific Language].

As it is a fully featured programming environment, Scripted Pipeline offers a tremendous amount of flexibility and extensibility to Jenkins users. The Groovy learning-curve isn't typically desirable for all members of a given team, so Declarative Pipeline was created to offer a simpler and more opinionated syntax for authoring Jenkins Pipeline.

The two are both fundamentally the same Pipeline sub-system underneath. They are both durable implementations of "Pipeline as code." They are both able to use steps built into Pipeline or provided by plugins. Both are able utilize Shared Libraries

Where they differ however is in syntax and flexibility. Declarative limits what is available to the user with a more strict and pre-defined structure, making it an ideal choice for simpler continuous delivery pipelines. Scripted provides very few limits, insofar that the only limits on structure and syntax tend to be defined by Groovy itself, rather than any Pipeline-specific systems, making it an ideal choice for power-users and those with more complex requirements. As the name implies, Declarative Pipeline is encourages a declarative programming model. [21: Declarative Programming] Whereas Scripted Pipelines follow a more imperative programming model.. [22: Imperative Programming]