Galaxy Tool Framework Changes

This document describes changes to Galaxy's tooling framework over recent releases.

16.04

Full Galaxy changelog.

Tool Profile Version (PR #1688)

Tools may (and should) now declare a `profile` version (e.g. `<tool profile="16.04" ...>`).

This allows Galaxy to fire a warning if a tool uses features too new for the current version and allows us to migrate away from some undesirable default behaviors that were required for backward compatiblity.

`set -e` by default (d020522)

From the IUC best practices documentation:

> *"If you need to execute more than one shell command, concatenate them with a double ampersand (`&&`), so that an error in a command will abort the execution of the following ones."*

The job script generated with profile `16.04` + tools will include a `#set -e` statement causing this behavior by default.

Older-style tools can enable this behavior by setting `strict="true"` on the tool `command` XML block.

Using Exit Codes for Error Detection (b92074e)

Previously the default behavior was for Galaxy to ignore exit codes and declare a tool in error if issued any output on standard error. This was a regretable default behavior and so all tools were encouraged to declare special XML blocks to force the use of exit codes.

For any tool that declares a profile version of `16.04` or greater, the default is now just to use exit codes for error detection.

Unrobust Features Removed (b92074e)

A few tool features have ben removed from tools that declare a version of `16.04` or newer.

- The `interepreter=` attribute on `command` blocks has been eliminated. Please use `$__tool_directory__` from within the tool instead.
- `format="input"` on output datasets has been eliminated, please use `format_source=` to specify an exact input to derive datatype from.
- Disables extra output file discovery by default, tools must explicitly describe the outputs to collect with `discover_dataset` tags.
- Tools require a `version` attribute - previously an implicit default to `1.0.0` would be used.
- `$param_file` has been eliminated.

Clean Working Directories

Previously, Galaxy would fill tool working directories with files related to metadata and job metric collection. Tools will no longer be executed in the same directory as these files.

This applies to all tools not just profile `16.04`+ tools.

16.01

Full Galaxy changelog.

Conda Dependency Resolution (PR #1345)

```
<tool>
    ...
    <requirements>
        <requirement type="package" version="0.11.4">FastQC</requirement>
    </requirements>
    ...
</tool>
```

- Dependency resolvers tell Galaxy how to translate requirements into jobs.
- The Conda dependency resolver forces Galaxy to create a conda environment for the job with `FastQC` at version `0.11.4` installed.
- Only dependency resolver that can be installed at runtime - great for Docker images, heterogeneous clusters, and testing tools.
- Links Conda and BioConda

ToolBox Enhancements - Labels (PR #1012)

ToolBox Enhancements - Monitoring (PR #1398)

- The Galaxy toolbox can be reloaded from the Admin interface.
- Tool conf files (e.g. `tool_conf.xml`) can be monitored and automatically reloaded by Galaxy.
- Tool conf files can now be specified as YAML (or JSON).

Process Inputs as JSON (PR #1405)

```
    <command>python "$__tool_directory/script.py" "$json_inputs"</command>
    <configfiles>
        <inputs name="json_inputs" />
    </configfiles>
```

This will produce a file referenced as `$json_inputs` that contains a nested JSON structure corresponding to the tools inputs. Of limitted utility for simple command-line tools - but complex tools with many repeats, conditional, and nesting could potentially benefit from this.

For instance, the JBrowse tool generates a complex JSON data structure using a `configfile` inside the XML. This is a much more portable way to deal with that.

Collections

- `data_collection` tool parameters (`param`s) can now specify multiple `collection_type`s for consumption (PR #1308).
  - This mirrors the `format` attribute which allows a comma-separated list of potential format types.
- Multiple collections can now be supplied to a `multiple="true"` data parameter (PR #805).
- Output collections can specify a `type_source` attribute (again mirroring `format_source`) (PR #1153).

15.10

Full Galaxy changelog.

Collections

- Tools may now produce explicit nested outputs PR #538. This enhances the `discover_dataset` XML tag to allow this.
- Allow certain `output` actions on collections. PR #544.
- Allow `discover_dataset` tags to use `format` instead of `ext` when referring to datatype extensions/formats.
- Allow `min` / `max` attributes on multiple data input parameters PR #765.

Whitelist Tools that Generate HTML (PR #510)

Galaxy now contains a plain text file that contains a list of tools whose output can be trusted when rendering HTML.

15.07

Full Galaxy changelog.

Parameterized XML Macros (PR #362)

Macros now allow defining tokens to be consumed as XML attributes. For instance, the following definition

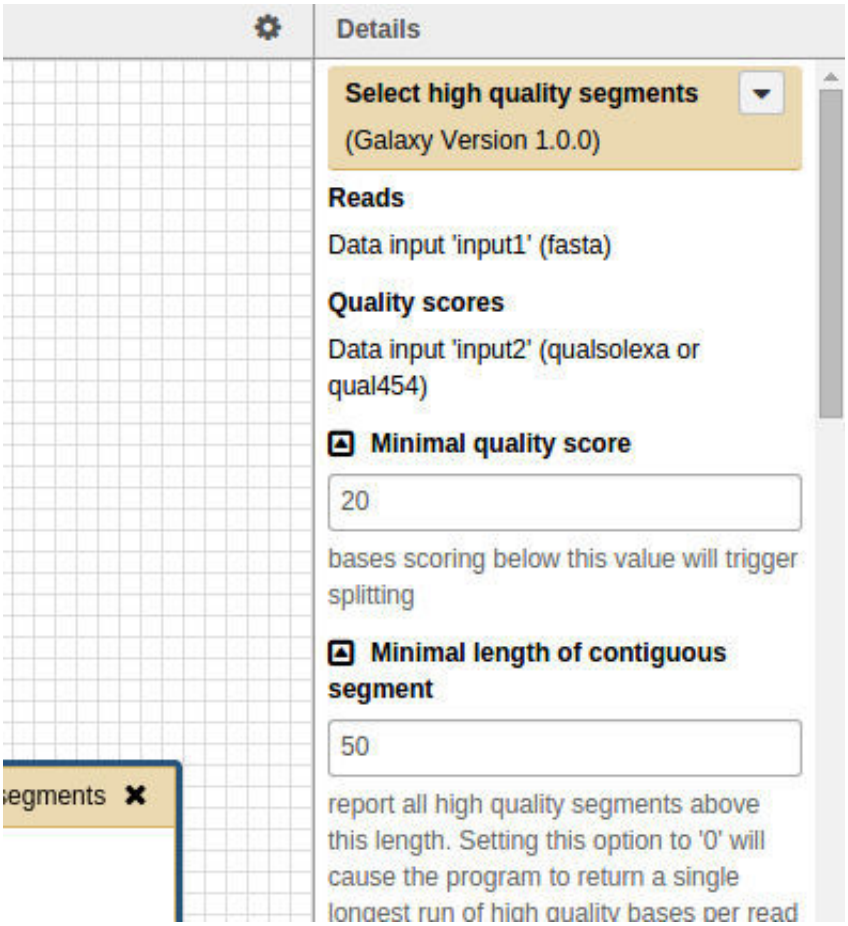
```
<tool>
    <expand macro="inputs" foo="hello" />
    <expand macro="inputs" foo="world" />
    <expand macro="inputs" />
    <macros>
        <xml name="inputs" token_foo="the_default">
            <inputs>@FOO@</inputs>
        </xml>
    </macros>
</tool>
```

would expand out as

```
<tool>
    <inputs>hello</inputs>
    <inputs>world</inputs>
    <inputs>the_default</inputs>
</tool>
```

Tool Form

The workflow editor was updated to the use Galaxy's newer frontend tool form.



Environment Variables (PR #395)

Tools may now use `inputs` to define environment variables that will be set during tool execution. The new `environment_variables` XML block is used to define this.

```
<tool>
    ...
    <command>
        echo "\$INTVAR"  >  $out_file1;
        echo "\$FORTEST" >> $out_file1;
    </command>
    <environment_variables>
        <environment_variable name="INTVAR">$inttest</environment_variable>
        <environment_variable name="FORTEST">#for i in ['m', 'o', 'o']#$i#end for#
</environment_variable>
    </environment_variables>
    ...
```

Test tool demonstrating the use of the `environment_variables` tag.

Collections

- Explicit output collections can now be used in workflows. (PR #311)
- The `filter` tag has been implemented for output dataset collections (PR #455. See the example tool output_collection_filter.xml.