Jenkins: AMPLab’s Friendly Butler

He will build your projects so you don’t have to!
What is Jenkins?

- Open source CI/CD/Build platform
- Used to build many, many open source software projects (including Spark and Tachyon)
- Jenkins CI homepage
What can Jenkins do for you?

Quite a bit, actually:

● Automatically build GitHub pull requests
● Test projects automatically
● Publish build artifacts for private or public consumption
● It doesn’t matter if it’s a Scala, Java or shell scripts: Jenkins can build anything!
When would I use Jenkins?

If any number of the items below are true:

- Your project has reached ‘critical mass’:
  - more than a couple contributors
  - many tests or branches
  - release to the public, even as alpha/beta

- You want to automate build and deployment duties:
  - publishing artifacts
Requirements for building a project

It’s pretty easy:

● Your project must be in GitHub.
  ○ Using private vs. public repos...
● Read the docs.
  ○ Learn to use Jenkins (specifically sections on building projects).
● Have a rough idea of what you want to do...
  ○ Run tests triggered from a GitHub pull request
  ○ Build different configurations of a project (for example, against different versions of Hadoop)
  ○ Build and publish artifacts
Let’s set up an example build!

What we’re going to do:

● Create a new build for a GitHub project.
● Set up the build:
  ○ GitHub repo url
  ○ Build triggers
  ○ Build actions (Maven goals, bash/shell execution)
  ○ Miscellaneous errata (build notifications, etc)
What’s the story?

Evan Sparks asked Shane:

“How can I give my users the most up to date version of my code (ML Matrix) automatically?”

Shane’s response:

“Let’s make a build that creates a jar, and uploads it to an Amazon S3 folder!”
What’s the story (part deux)?

We clarified the following before creating the build:

- How does it actually build?
  - SBT is included in the repo, so we just need to execute that with the proper arguments to create the jar artifact

- When do we want it to build?
  - Whenever new code is pushed to the GitHub repo

We also had to create a publicly readable S3 bucket for the artifact to publish to.
First, visit our very own Jenkins!
https://amplab.berkeley.edu/jenkins

...and log in with your EECS LDAP credentials!
We named the build, and chose the basic project configuration “Freestyle project”. Click ‘OK’ to continue.
Add a description.

Distributed linear algebra on Spark.

https://github.com/amplab/ml-matrix/
Add the URL to the GitHub project.

https://github.com/amplab/ml-matrix/
It’s good practice to put your email address here, so that you get notified if someone accidentally changes your job configuration.
Let Jenkins know which workers the job will run on. “centos” is the default for all builds.
Point the build to the GitHub repo. You don’t need credentials if it’s a public repo.
Decide how you want your build to trigger.
Let’s have it periodically poll GitHub for changes! But… how do I configure this?
Pro tip! Click on the little ? on the right of the field for inline help!
WOW! Many Helps!
Much wordiness!

This field follows the syntax of cron (with minor differences). Specifically, each line consists of 5 fields separated by TAB or whitespace:

- `MINUTE` Minutes within the hour (0-59)
- `HOUR` The hour of the day (0-23)
- `DOM` The day of the month (1-31)
- `MONTH` The month (1-12)
- `DOW` The day of the week (0-7) where 0 and 7 are Sunday.

To specify multiple values for one field, the following operators are available. In the order of precedence,

- `*` specifies all valid values
- `-` specifies a range of values
- `,` specifies a list of values
- `/` specifies a step interval

For example, `0 0 1 1-12 0-6` will match every minute, every hour, every 1st day of the month, from January to December, and every Sunday.

The `*` symbol can be used with a range. For example, `1-59 */3 * * 1` means some time between 12:00 AM (midnight) to 7:59 AM. You can also use step intervals with `*, or within ranges.

Beware that for the day of month field, short cycles such as `/1` or `*/3` will not work consistently near the end of most months, due to variable month lengths. For example, `*/3` will run on the 1st, 4th, ..., 31st days of a long month, then again the next day of the next month. Hashes are always chosen in the 1-31 range, so `1/3` will produce a gap between rows of between 3 and 8 days at the end of a month. (Longer cycles will also have inconsistent lengths but the effect may be relatively less noticeable.)

Empty lines and lines that start with `#` will be ignored as comments.

In addition, `@yearly`, `@annually`, `@monthly`, `@weekly`, `@daily`, and `@hourly` are supported as convenient aliases. These use the hash system for automatic balancing. For example, `@hourly` is the same as `0 * * * *` and could mean at any time during the hour. `@midnight` actually means some time between 12:00 AM and 1:59 AM.

Examples:

- `# every Fifteen minutes (perhaps at 07, 12, 17, 22)`
- `H/15 * * *`
- `# every 10 minutes in the first half of every hour (three times, perhaps at 04, 014, 024)`
- `H(0-29)/10 * * *`
- `# once every two hours every weekday (perhaps at 10:38 AM, 12:38 PM, 2:38 PM, 4:38 PM)`
- `M 9-16/2 * * 1-5`
- `# once a day on the 1st and 15th of every month except December`
- `H 1,15 1-11`
Let's just keep it simple and modify one of the given examples: Poll every 5 minutes.
Enter the cron pattern in the text box.

Pro tip: click the ? to close the help dialogue!
It’s good practice to have the build abort if it’s stuck. This build will typically take <1 min, so let’s set the timeout to 10 minutes.
We also like pretty colors in our console output. It makes debugging easier!
Now we define the build steps. Here, since SBT is included in the repo, we choose “Execute shell”.
Enter the command to build the jar.

Command: `abt/abt package`
Next, we will add a Post-build action to publish the artifact.
Click ‘Add’ on Files to define your upload.
Fill in these bits (which Shane will help you with, based on the S3 bucket you’d be publishing to).
And click Save!
Things to keep in mind...

- There are a million little options, but most jobs only need a few things:
  - GitHub url, polling behavior, build execution
- Look at other builds for ideas on how to configure and create your own.
- Don’t be afraid to experiment!
- If you break it, we’ll be able to fix it. :)
- Asking me for help and advice is encouraged!