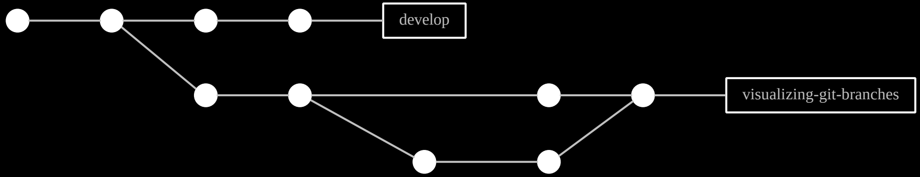
Git DAGs

Correl Roush

December 2, 2015

Extreme Tech Seminar

VISUALIZING GIT BRANCHES



THE IDEA

Digging through Derek Feichtinger's org-babel examples (which I came across via irreal.org), I found he had some great examples of displaying git-style graphs using graphviz. I thought it'd be a fun exercise to generate my own graphs based on his graphviz source using elisp, and point it at actual git repos.

DAGS

DIRECTED ACYCLIC GRAPHS

In mathematics and computer science, a directed acyclic graph (DAG), is a directed graph with no directed cycles. That is, it is formed by a collection of vertices and directed edges, each edge connecting one vertex to another, such that there is no way to start at some vertex v and follow a sequence of edges that eventually loops back to v again.
– Wikipedia

GIT AS A DAG

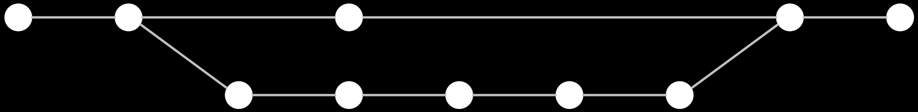
http://eagain.net/articles/git-for-computer-scientists/

BUILDING GRAPHS

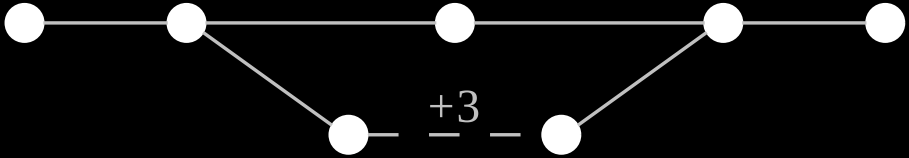
http://correl.phoenixinquis.net/2015/07/12/git-graphs.html

COLLAPSING HISTORY

Collapsing this:



To this:



Something like this would be handy for concisely graphing the state of multiple ongoing development branches (say, to get a picture of what's been going on since the last release, and what's still incomplete).

HOW TO COLLAPSE HISTORY

- Determine what portions of a graph aren't worth drawing
  - Stretches of non-parent, non-parented commits
  - Merged branches (except branches we always want to show)
- Determine how to apply a set of criteria to a graph
  - Can it be done w/o scanning the whole graph?

Final thoughts