

Distributed Version Control Systems Git

NSCoderNight London Sep 2010

Alex Blewitt @alblue

<http://alblue.bandlem.com>



Centralised VCS

RCS

1.1

1.2

1.3

CVS

1.1

1.2

1.3

1.4

1.5

1.1

1.2

1.3

Svn

r1

r2

r4

r5

r1

r2

r3

r5

Central Server

- Keeps track of revision numbers
 - Per file revision for CVS
 - Per repository version for SVN
- Mediates multiple users
- Must be on-line to query state
- May perform (pessimistic) file locking

Distributed Server?

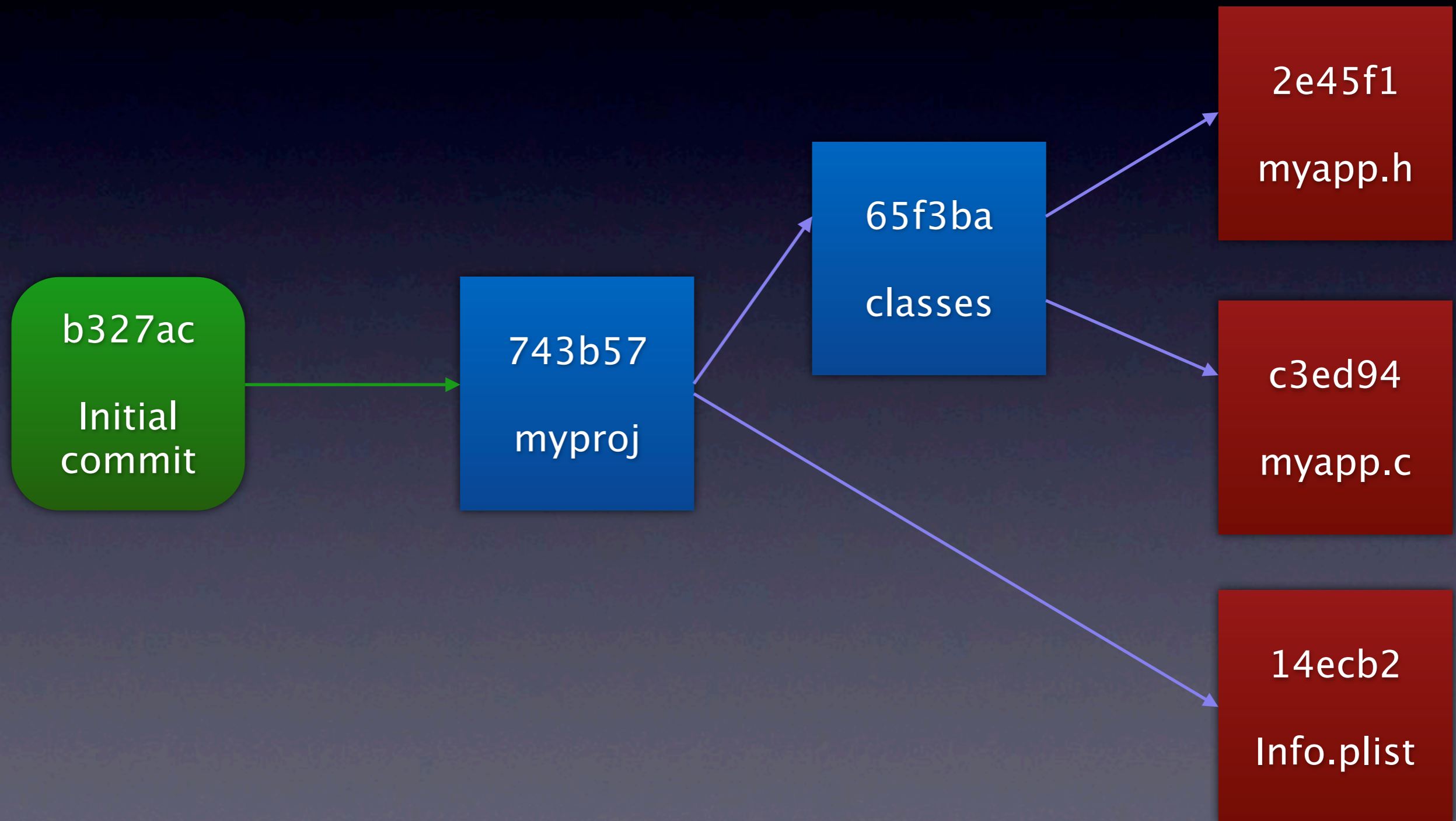
- No way of enforcing global versions
 - If we both change to 1.2, what is it called?
- No way to perform locking
- No central server* so everyone has a copy

*though you can make one copy a central one if you want

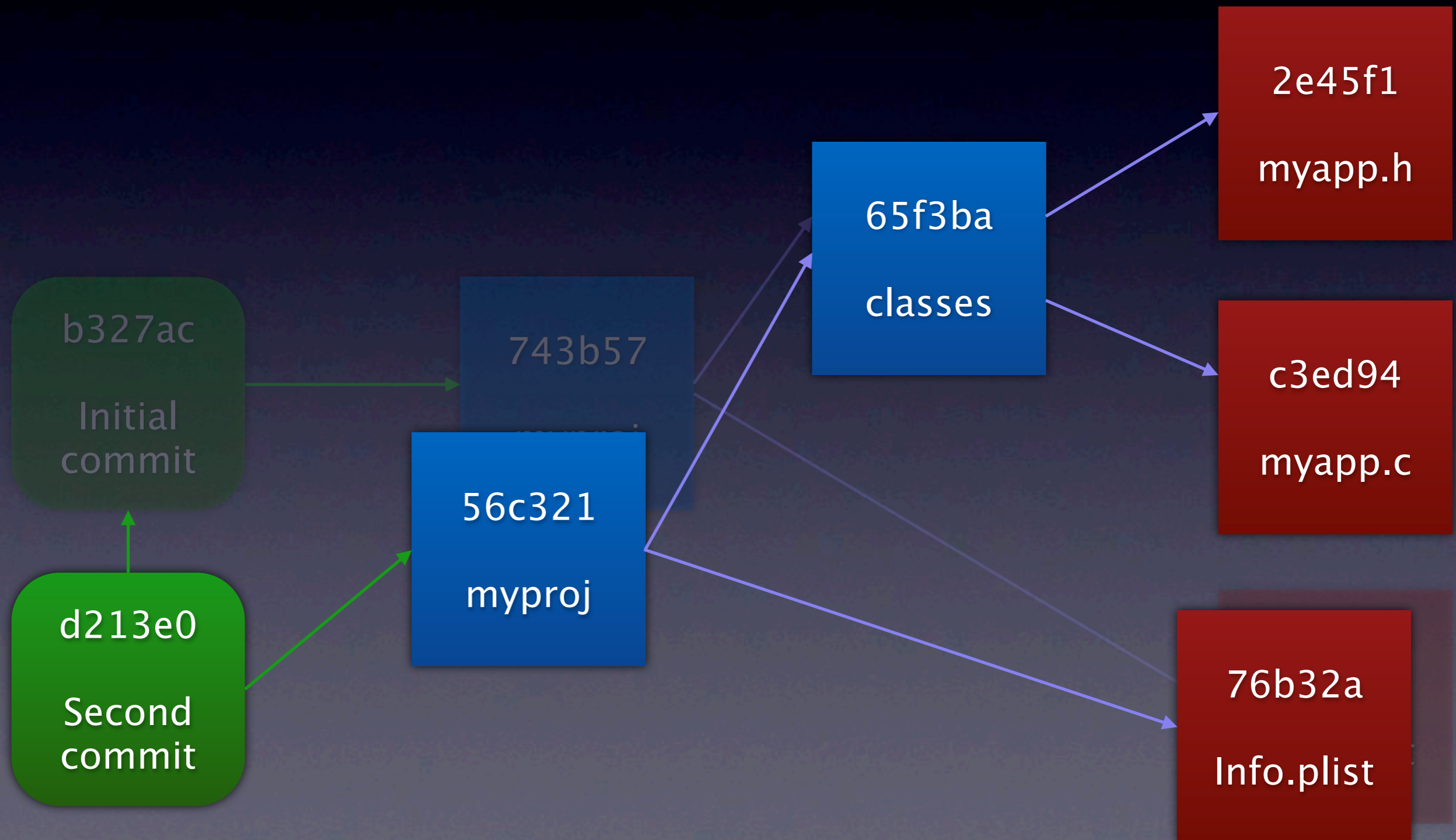
Naming revisions

- Globally incrementing numbers won't work
- Contents are always same though
 - So use (hash of) contents for identification
- SHA-1 hash frequently used for DVCS

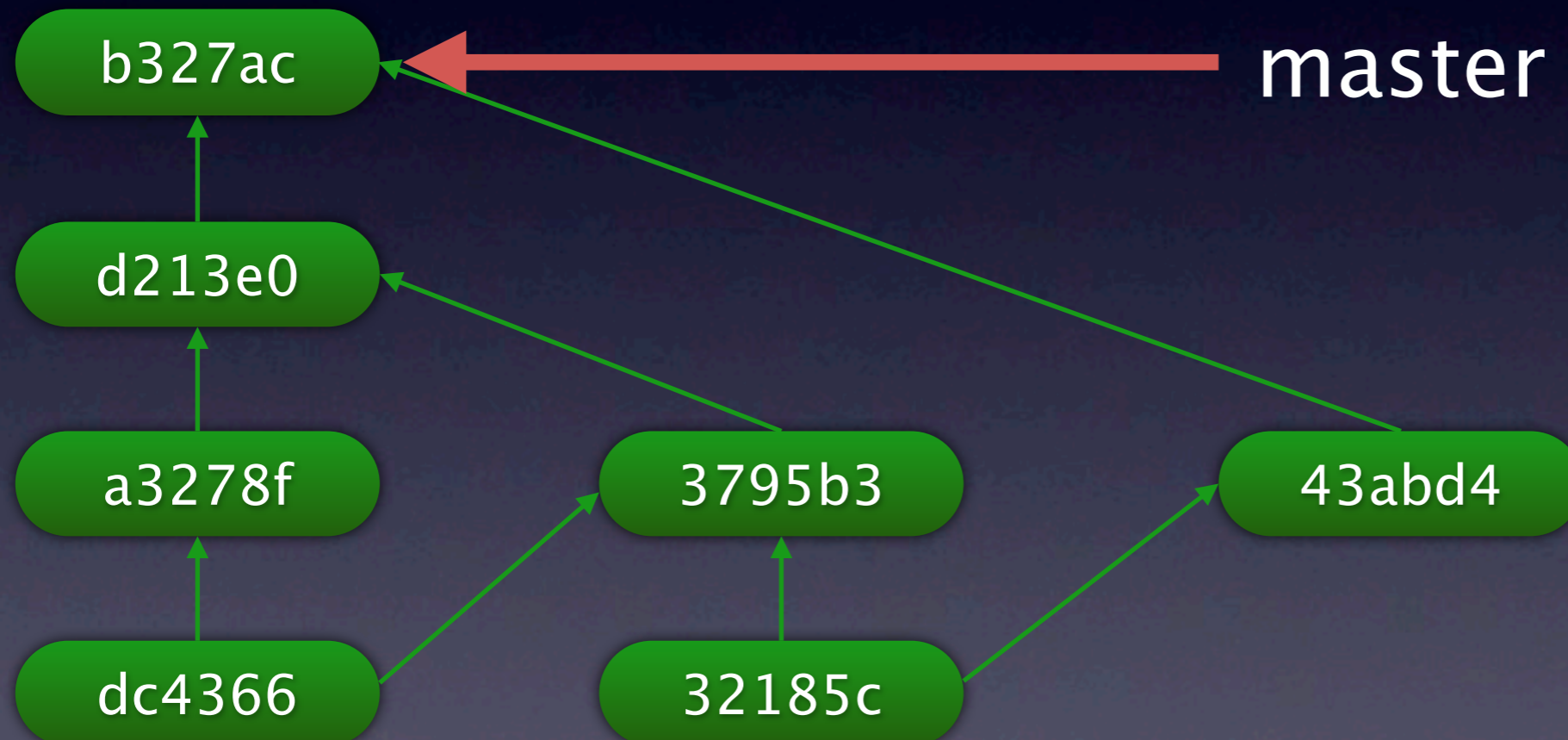
Git object hierarchy



Git object hierarchy



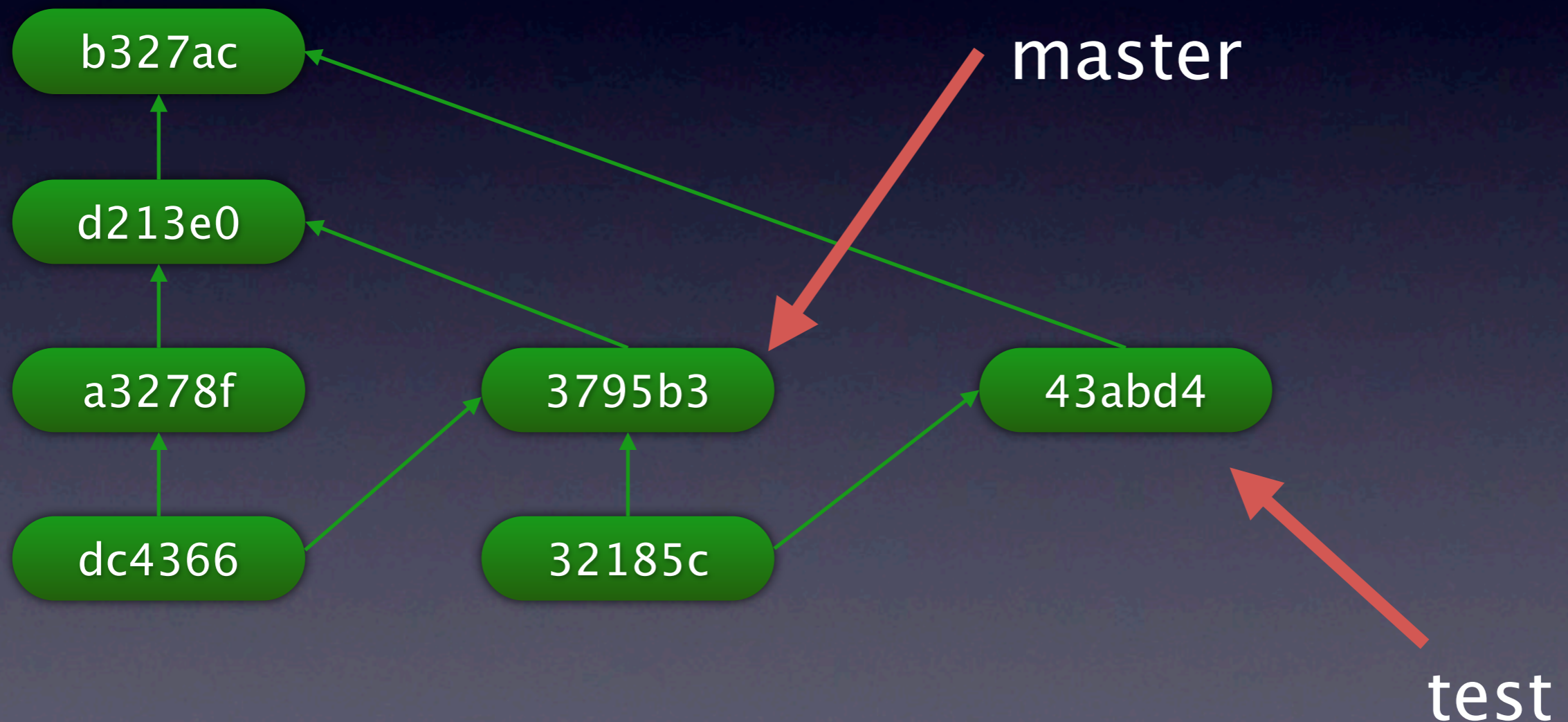
Git branches



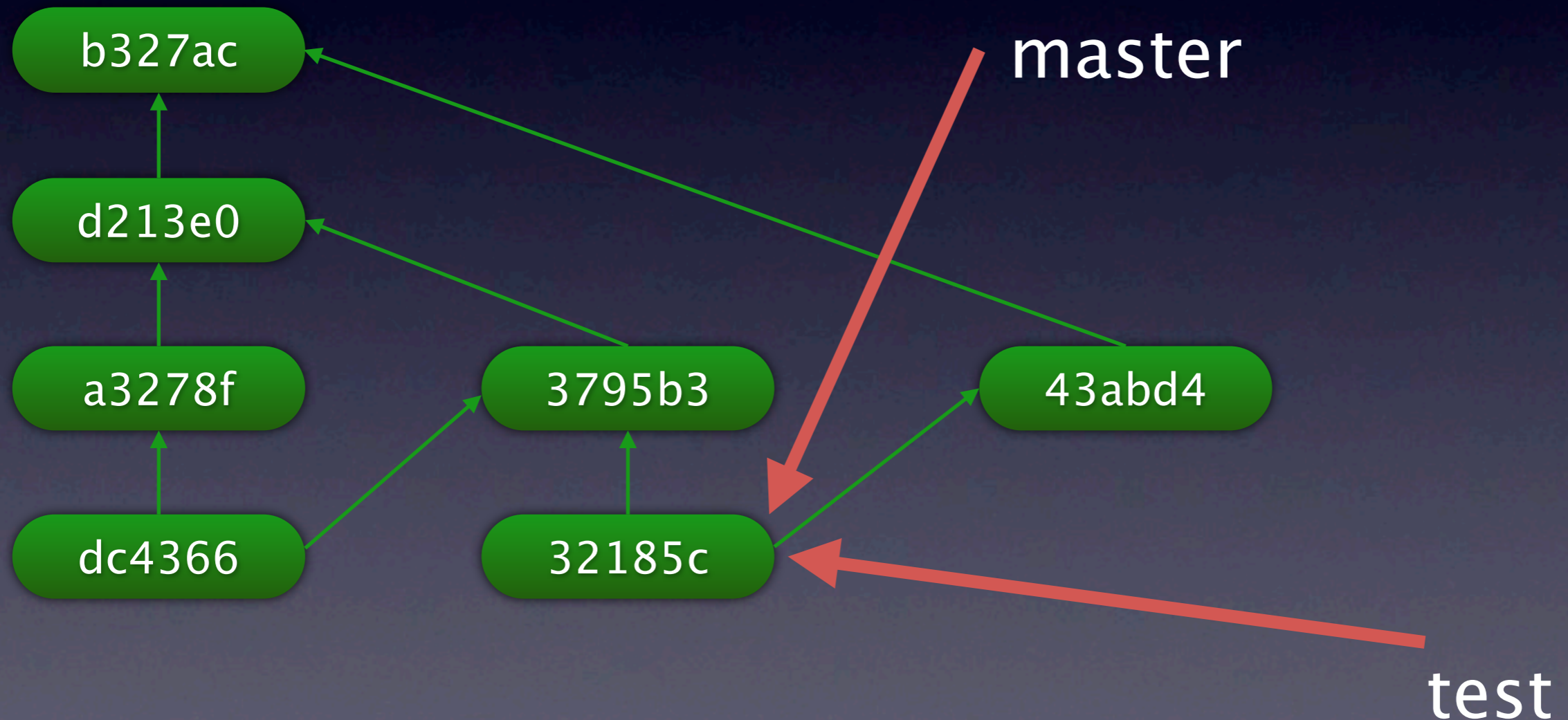
Git branches



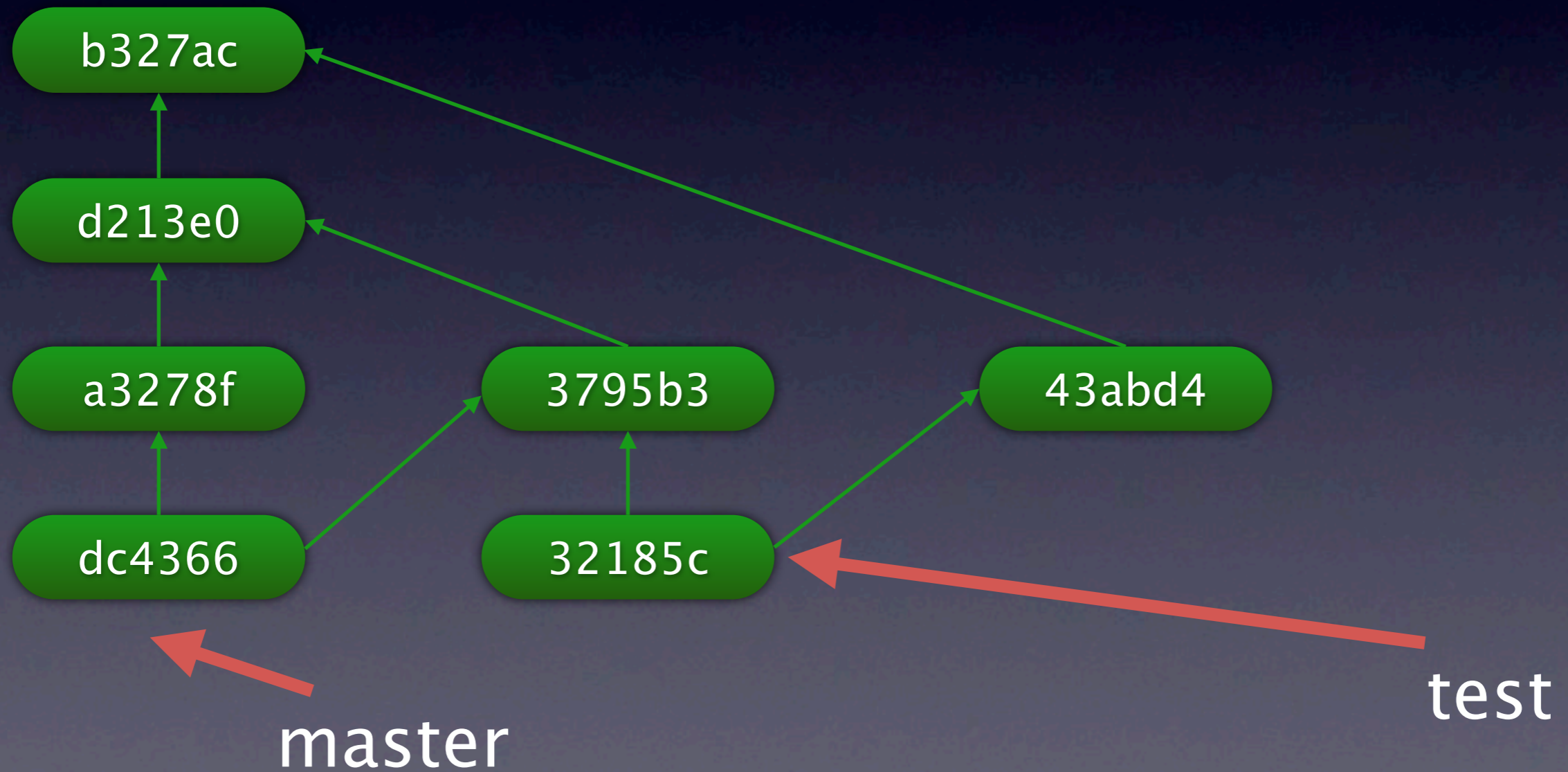
Git branches



Git branches



Git branches



Git branches

- Branches are cheap
- Merging is trivial — the branch point is known
- Branches created and destroyed frequently
 - Adding a new feature
 - Experimentation
 - Per-bug fixes

Git distribution

- `git clone` — take a copy of remote repository
- `git fetch` — get changes from remote
- `git pull` — merge changes from remote
- `git push` — send changes back to remote

Getting started

- `git init` — creates a repository
- `git commit (-a)` — adds (everything) to repo
- `git checkout -b test` — switches to branch test
- `git branch` — see what branches there are
- `git log` — show history
- `git status` — show uncommitted/changed files

Identifying treeish

- Commits have an SHA1 hash
 - Can refer to them with any unique prefix
 - Typically six characters are used
- Can use (remote) branch names or tag names

Other references

- HEAD refers to the last commit
 - Not the same as SVN's trunk or CVS' HEAD
- HEAD^^^ or HEAD~3 refer to grandparent
- HEAD@{3} is what it was 3 changes ago
- HEAD@{yesterday} == HEAD@{1.day.ago}

Ranges

- Some commands take ranges (diff,log,merge)
 - start..end — commits in end not start
 - start...end — commits in end and start
- Some commands can take multiple treeish
 - git log start ^end middle -not earth

Git concepts

- Reset — move branch to different commit
- Merge — merge two+ commits into tree
- Rebase — modify order of commits
- Squash — merge two+ commits into one
- Amend — redo last commit

Advantages of git

- GitHub
- Speed
- Ability to rewrite commits
- And standard DVCS features...
 - All history is local
 - Can operate without network

Git power tools

Git bisect

- Binary search for some condition
- Define “good” test and “bad” test
- Recursively checks out code until commit hit

Git filter-branch

- Can rewrite any commit in history
- Used for removing sensitive data/files
- Can be used to split trees
- Changes every SHA — use with caution!

Git submodules

- Can create a “parent” git repository
- Points to specific commits in submodules
- Must push submodules as well as parent
- Not widely used

Git and the index

- Git has an intermediary stage of committing
- Can build up a change set interactively
- Can commit everything with `git commit -a`
- Can “stash” changes for later
- Power tool whose use isn't initially obvious

Introspecting the data

- `git show` — show the object type
- `git ls-tree` — show contents of tree
- `git cat-file` — show contents of blob

Distributed Version Control Systems Git

Made on an iPad 

NSCoderNight London Sep 2010

Alex Blewitt @alblue

<http://alblue.bandlem.com>

