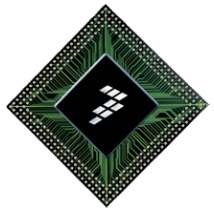
August 4, 2008

Putting the IT in Git



Jon Loeliger
Software Engineer

freescale™ semiconductor

Putting the IT in Git: Outline

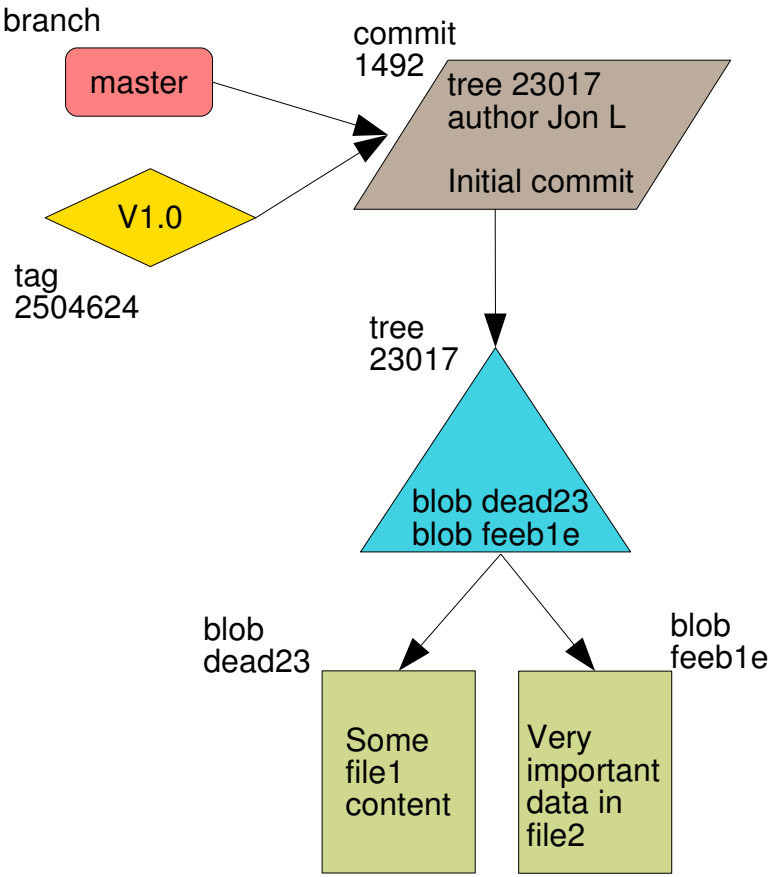
- Introduction to Git
  - Concepts
  - Basic Git
- Using Hooks
  - Pre-commit Validation Example
  - Post-receive Publishing Example
- Really Slick Git Projects
  - gitosis by Tommi Vertanen
  - etckeeper by Joey Hess
  - ikiwiki by Joey Hess

Introduction: Git

- Peer-to-peer Distributed Version Control System
- Scalability
- Fast and efficient
- Atomicity
- Verifiable repository integrity
- Immutability
- On-line and off-line development
- Open License

Concepts: Git Objects

- Objects
  - **Blob** – Raw file data
  - **Tree** – One directory level
    - Blob list (files)
    - Nested trees (sub-directories)
  - **Commit** – Repository snapshot at some instant in time
    - One or more Trees
    - Log message
    - Author and date
  - **Tag** – Signed commit identifier
    - Commit
    - Tagger
    - GPG Signature



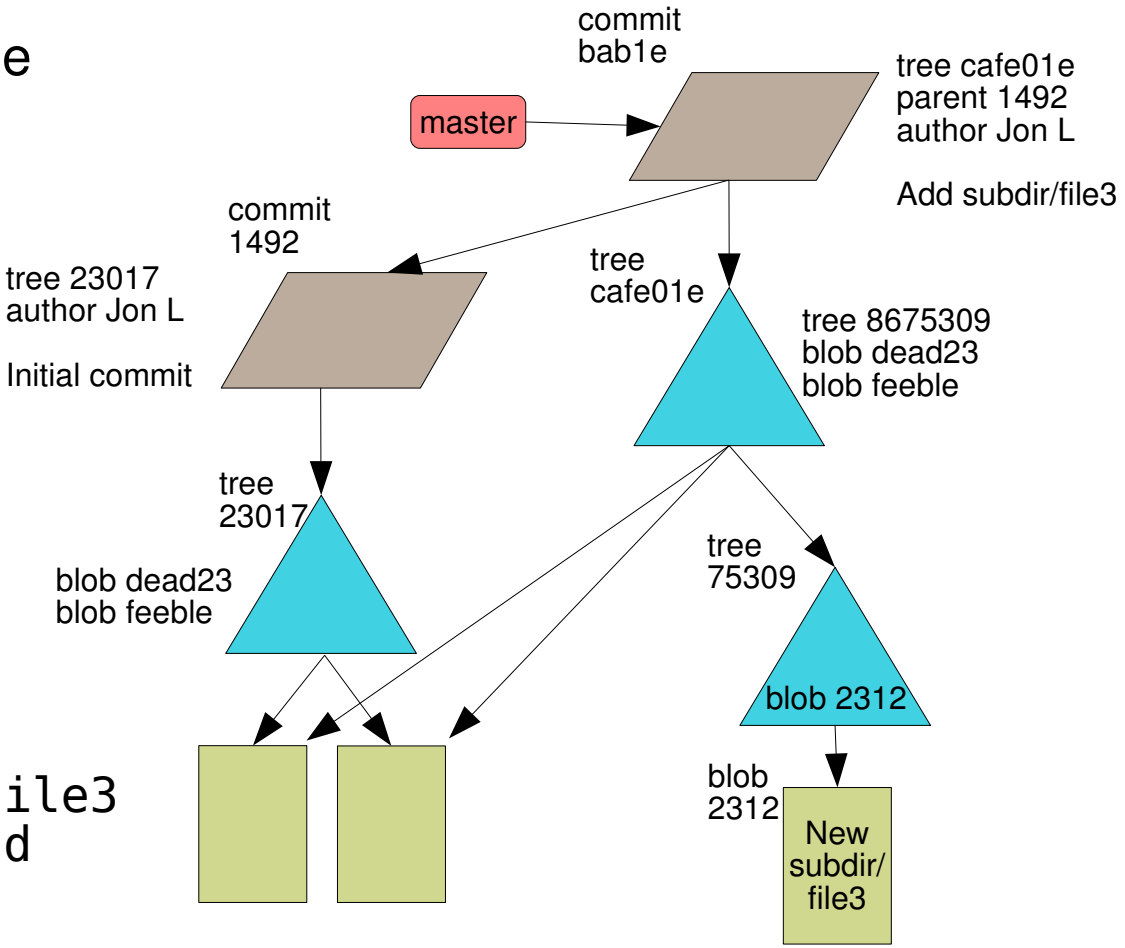
- Add two files

```
$ git add file1 file2
$ git commit -m “Initial
  commit”
```


freescale™ semiconductor

Concepts: Content Addressable Store

- Shared objects
  - Content Addressable Objects Store



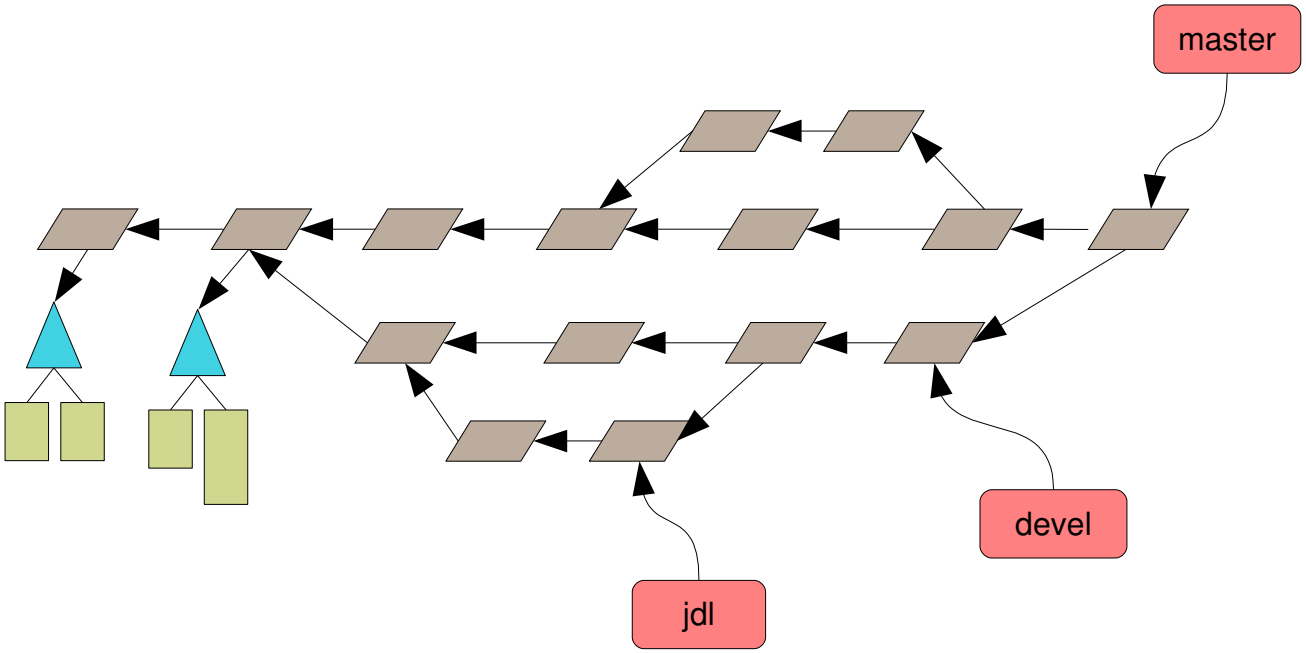
- Add new file

```
$ git add subdir/file3
$ git commit -m”Add
  subdir/file3”
```


freescale™ semiconductor

Concepts: Commit Graph Implementation

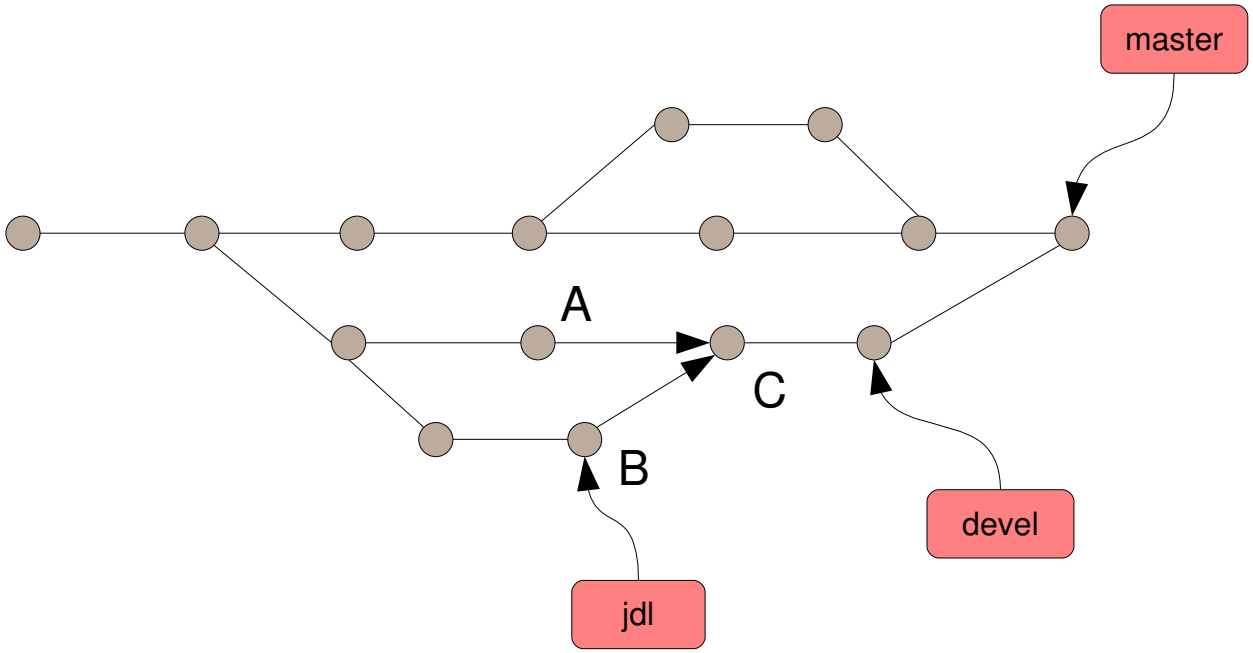
- Commit Graph
  - Directed Acyclic Graph
  - Commits point back to their parent commits



freescale semiconductor

Concepts: Commit Graph Simplified

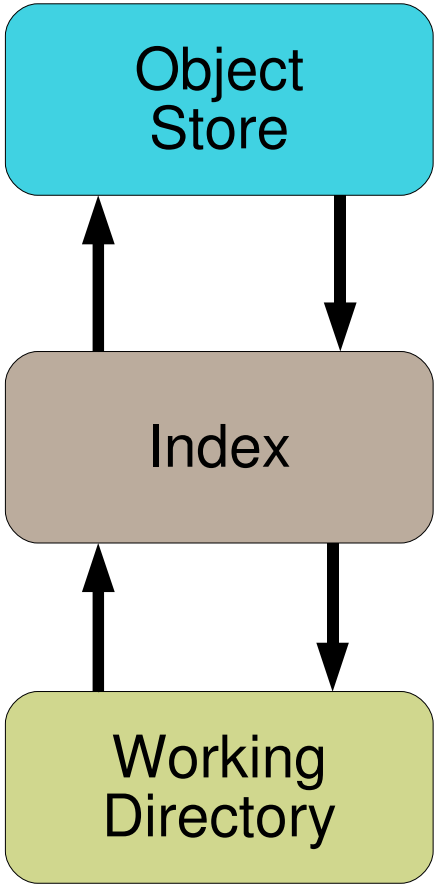
- Commit Graph Redux
  - Sequence of commit nodes over time
  - Time flows left-to-right, loosely
  - Apparent “data flow” drawn left-to-right also!
    - A and B are merged into C



freescale semiconductor

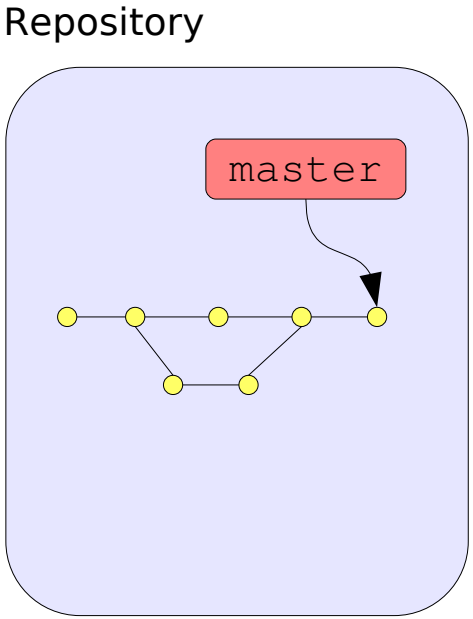
Concepts: Index

- Index is a staging area for tree objects
  - Collects file and directory changes
  - Create new tree objects
  - Holds current or past tree objects
- Index holds temporary objects
  - Helps resolve merge conflicts



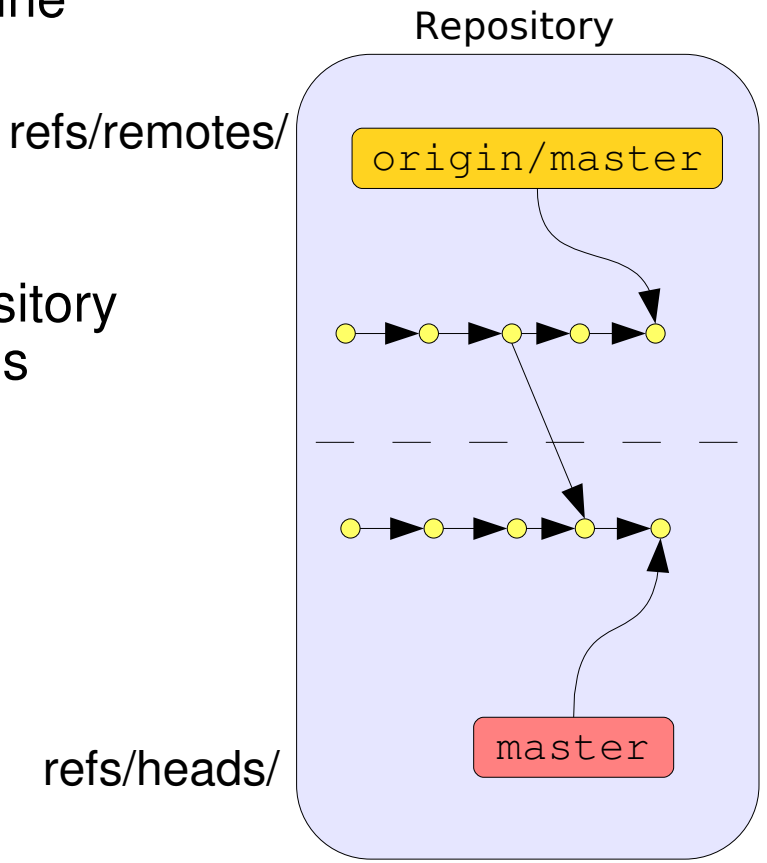
Concepts: Repositories

- Self-contained, local and complete
  - Repository data is not distributed
  - Supports both on- and off-line development
  - Includes complete history
- Distributed or centralized development model
  - Only authoritative by convention and agreement
  - Multiple developers work on common repository
- Bare versus Development Repository



Concepts: Branches

- Branches are names that point to a commit
  - Name the HEAD of a development line
  - Fast, cheap
- Tracking Branches
  - Follow changes from a remote repository
  - Not for your development or changes
  - Use refs/remotes/ name-space
- Topic Branches
  - Your local development
  - Use the refs/heads name-space



Basic Git: Create A New Repository

- Create a new repository

```
$ cd /some/directory
$ git init
```

- Establish Commit Author

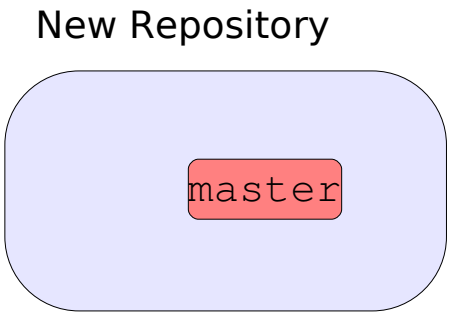
```
$ git config user.name 'Jon Loeliger'
$ git config user.email 'jdl@example.com'
```

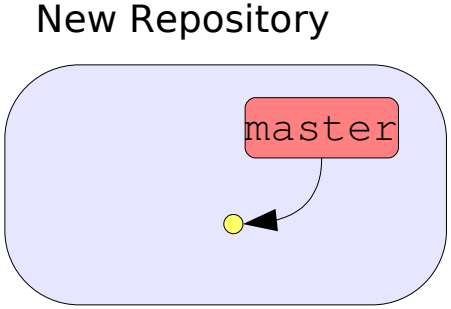
- Add new or existing files or updates

```
$ git add path/to/file
$ git add .
```

- Make first commit on `master` branch

```
$ git commit -m 'Initial commit!'
```

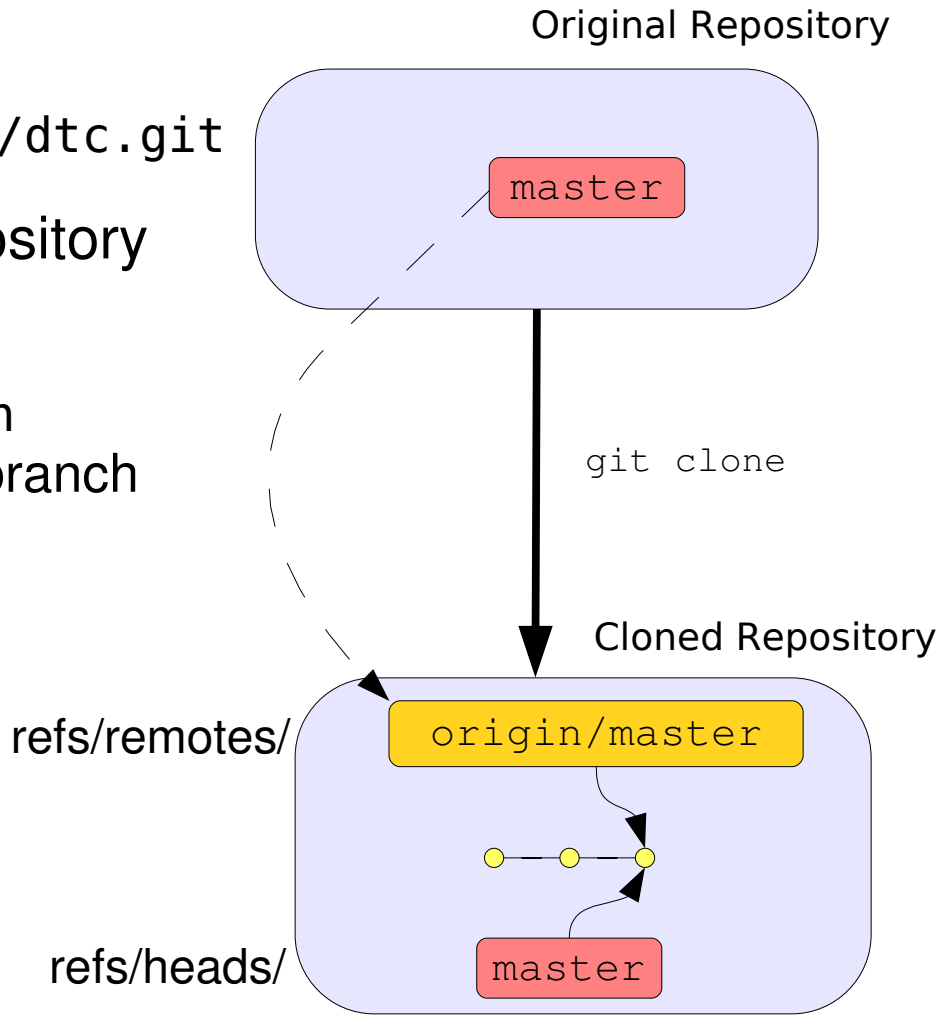




freescale™ semiconductor

Basic Git: Cloning

- Clone from upstream URL
  ```
  $ git clone
    git://www.jdl.com/software/dtc.git
  ```
- Copies complete upstream repository into a local repository
- Creates a *remote* named `origin`
  - Adds `origin/master` tracking branch
- Introduces `master` topic branch
  - Started at `origin/master`
  - For *your* development



freescale semiconductor

Basic Git: Edit Cycle

- `$ cd some/project`
- `$ git checkout master` `# Or your topic branch`
- `$ emacs some/file` `# Your favorite editor`
- `$ git status` `# Look around...`

```
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       WineLessons.html
nothing added to commit but untracked files present
 (use "git add" to track)
```

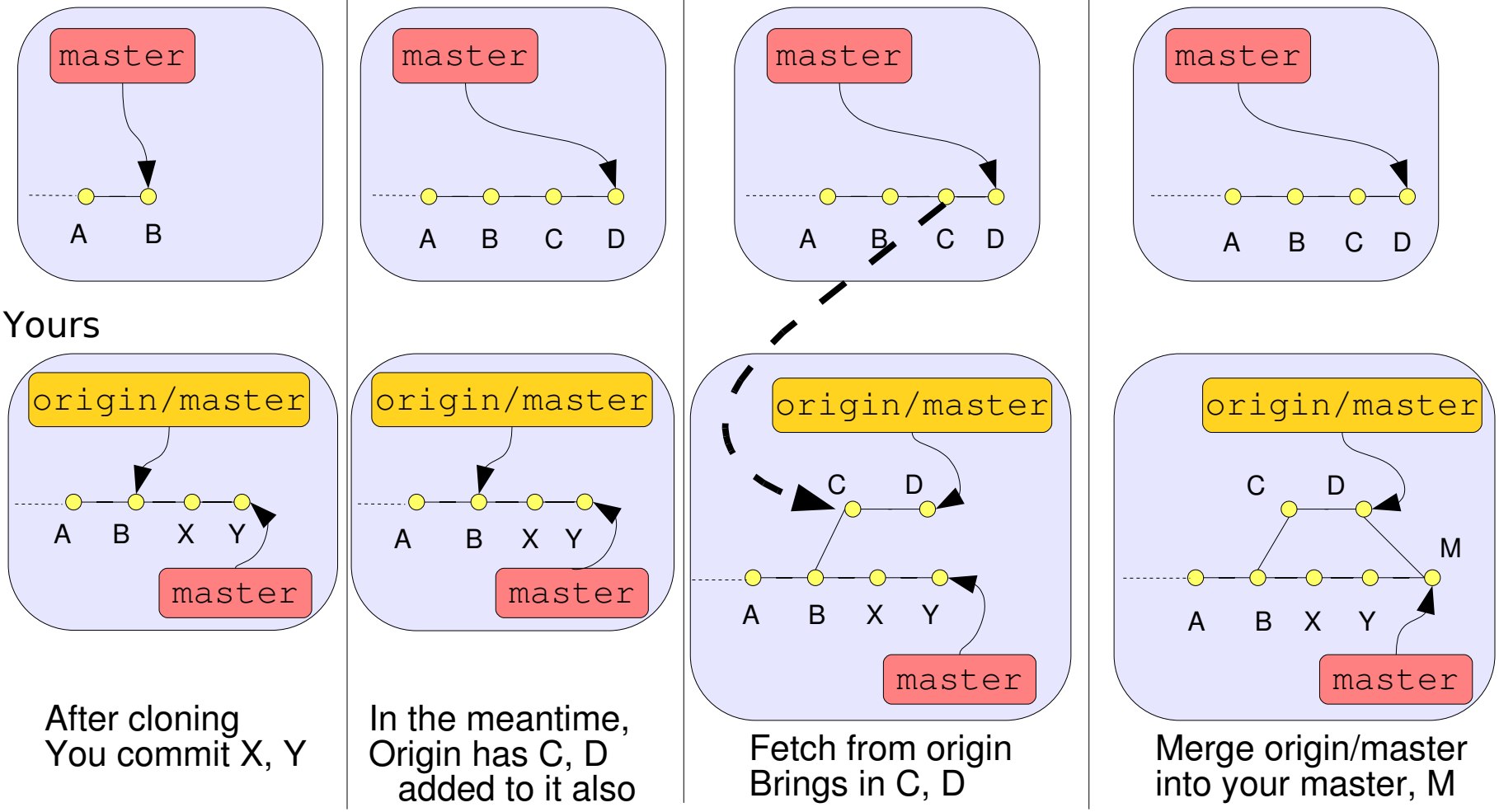
- `$ git add WineLessons.html` `# New or existing!`
- `$ git commit -m ”Quick log message here.”`

```
Created commit ee1f00d: Quick Log Message Here.
 1 files changed, 4 insertions(+), 1 deletions(-)
```

Basic Git: Fetch plus Merge

Origin

Yours



After cloning
You commit X, Y

In the meantime,
Origin has C, D
added to it also

Fetch from origin
Brings in C, D

Merge origin/master
into your master, M

freescale semiconductor

Git Hooks

- Git Hooks
  - Trigger actions at well-know times
  - Configured per-repository
  - You write the action script!

- Example Hooks:
  - **pre-commit** – Run prior to commit in local repository
    - Validate commits prior to check-in
  - **post-commit** – Run after commit in local repository
    - Perform action such as notification after a commit
  - **post-checkout** – Run after a branch is checked out
    - Perhaps update working directory components
  - **pre-receive** – Run in remote repository prior to receiving pushed data
    - Validate reception of incoming pushed data
  - **post-receive** – Run in remote repository after receiving pushed data
    - Send email notification after pushed data is received

Hooks: pre-commit Validation

- Using the pre-commit hook
  - Runs prior to commit in the local repository
  - Validate or reject commits
- Repository source files
  - DocBook articles written in XML
  - Makefile with build rules to convert XML to HTML
- Pre-commit hook uses “make” return code
  - Allow only valid XML to be committed
- Example using the hook
  - Show valid commit
  - Show invalid commit
  - Override validation check

Hooks Example: Source Files

- $ cd Articles
- $ ls -lsa

```
200K -rw-r--r--  docbook.dtd
  4K -rw-r--r--  Makefile
  4K -rw-r--r--  WineLessons.xml
```

- $ cat WineLessons.xml

```
<!DOCTYPE article PUBLIC "-//OASIS//DTD DocBook XML V5.0//EN"
  "docbook.dtd">
<article xmlns="http://docbook.org/ns/docbook">
<info>
  <title>Home Winemaking Lessons</title>
  <author><personname>
    <firstname>Jon</firstname><surname>Loeliger</surname>
  </personname></author>
</info>
<para>
Here are some lessons I learned about making wine at home.
</para>
<!-- Write juicy details! -->
</article>
```

Hooks Example: Build Files

- $ cat Makefile

```
PUBLIC_WEB      = ~/public_html/articles

ARTICLES        = WineLessons.xml
ARTICLES_HTML   = $(ARTICLES:.xml=.html)

all: $(ARTICLES_HTML)

%.html: %.xml
        xmlto xhtml-nochunks $<

clean:
        rm -rf $(ARTICLES_HTML)

install: $(ARTICLES_HTML)
        install $(ARTICLES_HTML) $(PUBLIC_WEB)
```

- DTD Source:

```
$ wget http://docbook.org/xml/5.0/dtd/docbook.dtd
```

Hooks Example: Initialize Repository

- Create Git repository

```
$ cd directory/with/Articles
$ git init
   Initialized empty Git repository in Articles/.git/
```

```
$ git add .
```

```
$ git commit -m "Add Wine Lessons article"
   Created initial commit 344058c: Add Wine Lessons article
    3 files changed, 4227 insertions(+), 0 deletions(-)
    create mode 100644 Makefile
    create mode 100644 WineLessons.xml
    create mode 100644 docbook.dtd
```

Hooks Example: pre-commit Script

- Pre-commit hook

```
$ cat .git/hooks/pre-commit

   #!/bin/sh
   if ! make ; then
       echo "You need to fix the build errors first."
       exit 1
   fi
   exit 0
```

- Enable the hook

```
$ chmod +x .git/hooks/pre-commit

$ ls -ls .git/hooks/pre-commit
   4K -rwxr-xr-x  .git/hooks/pre-commit
```

Hooks Example: Make a Good Edit

- Edit WineLessons.xml
  - Add a new section

```
$ git diff
   diff --git a/WineLessons.xml b/WineLessons.xml
   index da34189..a6b2bf9 100644
   --- a/WineLessons.xml
   +++ b/WineLessons.xml
   @@ -9,5 +9,8 @@
    <para>
    Here are some lessons I learned about making wine at home.
    </para>
   -<!-- Write juicy details! -->
   +<section xml:id="TheEasyWay">
   +<title>Lessons Learned the Easy Way</title>
   +<para> ... </para>
   +</section>
    </article>
```

Hooks Example: Use pre-commit Hook

- Commit using pre-validation

```
$ git commit -a -m"Add Easy Lessons"
   xmlto xhtml-nochunks WineLessons.xml
   Stripping NS from DocBook 5/NG document.
   Processing stripped document.
   Created commit e9ec498: Add Easy Lessons
    1 files changed, 4 insertions(+), 1 deletions(-)
```

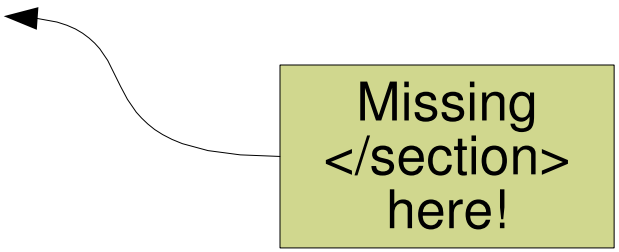
- Inspect the log

```
$ git log
   commit e9ec498cdf2d93e13ccc7c73056f2d486235513d
   Author: Jon Loeliger <jdl@example.com>
   Date:   Thu Jun 5 16:37:26 2008 -0500

       Add Easy Lessons

   commit 344058ccb70799350a05bbe022e7b25d9b226f8d
   Author: Jon Loeliger <jdl@example.com>
   Date:   Thu Jun 5 16:25:37 2008 -0500

       Add Wine Lessons article
```

Hooks Example: Make a Bad Edit

- Edit WineLessons.xml
  - Add a new section with typo

```
$ git diff
   diff --git a/WineLessons.xml b/WineLessons.xml
   index a6b2bf9..94011cc 100644
   --- a/WineLessons.xml
   +++ b/WineLessons.xml
   @@ -13,4 +13,8 @@ Here are some lessons I learned about making
     wine at home.
    <title>Lessons Learned the Easy Way</title>
    <para> ... </para>
    </section>
   +<section xml:id="TheHardWay">
   +<title>Lessons Learned the Hard Way</title>
   +<para> ... </para>
   +
    </article>
```



Hooks Example: pre-commit Rejection

- Commit using pre-validation

```
$ git commit -a -m"Add Hard Lessons"
   xmlto xhtml-nochunks WineLessons.xml
   xmlto: input does not validate (status 1)
   /tmp/Articles/WineLessons.xml:20: parser error : Opening and
     ending tag mismatch: section line 16 and article
   </article>
             ^
   /tmp/Articles/WineLessons.xml:21: parser error : Premature end of
     data in tag article line 2
   ^
   make: *** [WineLessons.html] Error 1
   You need to fix the build errors first.
```

- Override pre-commit validation

```
$ git commit --no-verify -a -m"Add Hard Lessons"
   Created commit 215889a: Add Hard Lessons
    1 files changed, 4 insertions(+), 0 deletions(-)
```

Hooks Example: Ignoring Files

- The `.gitignore` file
  - Used to exclude path-name patterns
    - `*.html` or `*.o`
  - Exclude generated or temporary files
  - In any directory of your project
  - Ignored files won't be cloned or committed
  - Project wide, including all clones
- Use `.git/info/exclude`
  - Repository specific exclusion rules
  - Rules not copied during clone

Hooks Example: Ignore Generated Files

- Noise due to generated file

```
$ git status
   # On branch master
   # Untracked files:
   #  (use "git add <file>..." to include in what will be committed)
   #
   #       WineLessons.html
```

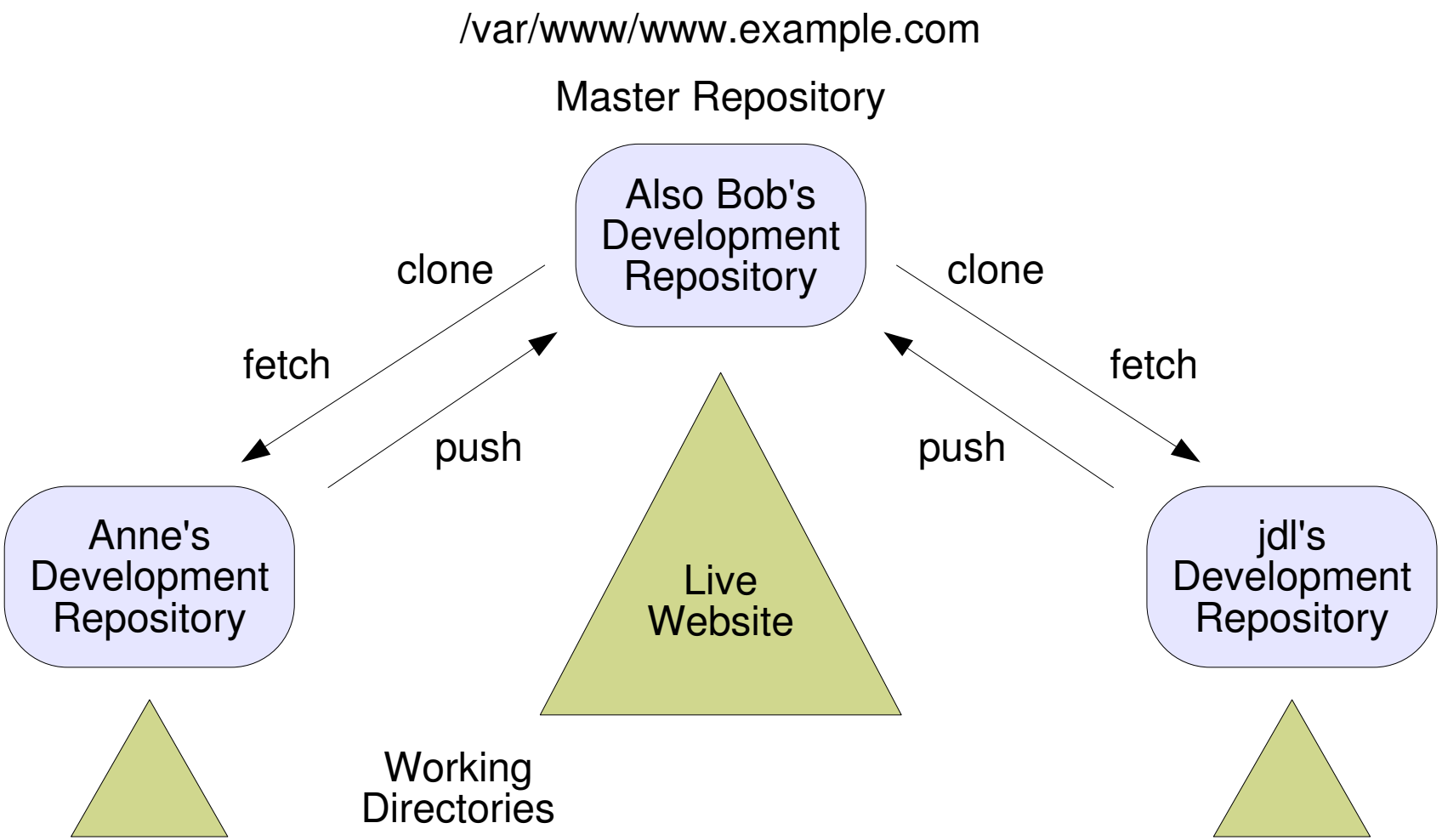
- Create `.gitignore` file

```
$ echo '*.html' > .gitignore
$ git add .gitignore
$ git commit -m “Ignore generated files”
```

- Noise excluded

```
$ git status
   # On branch master
            nothing to commit (working directory clean)
```

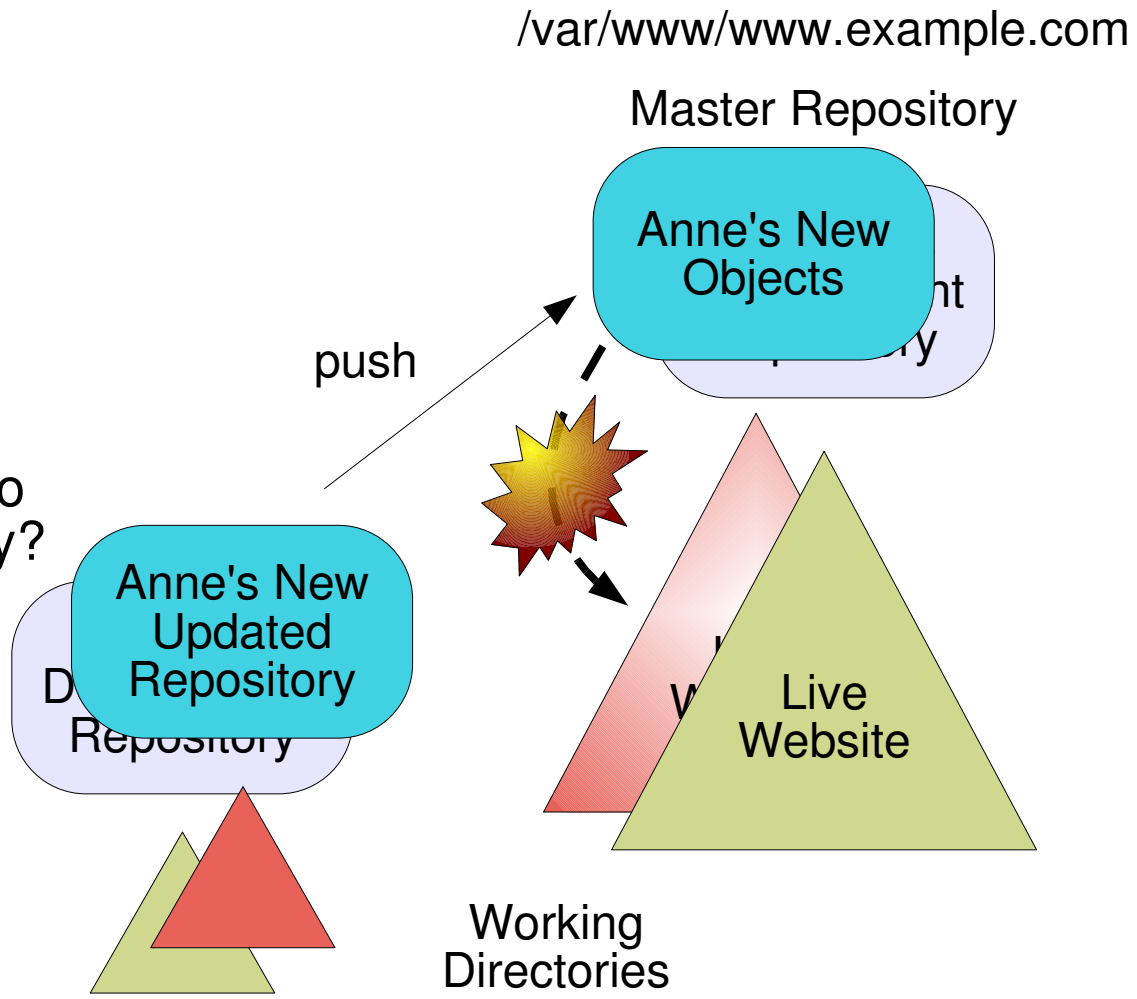
Example: Website Developed In Place



freescale™ semiconductor

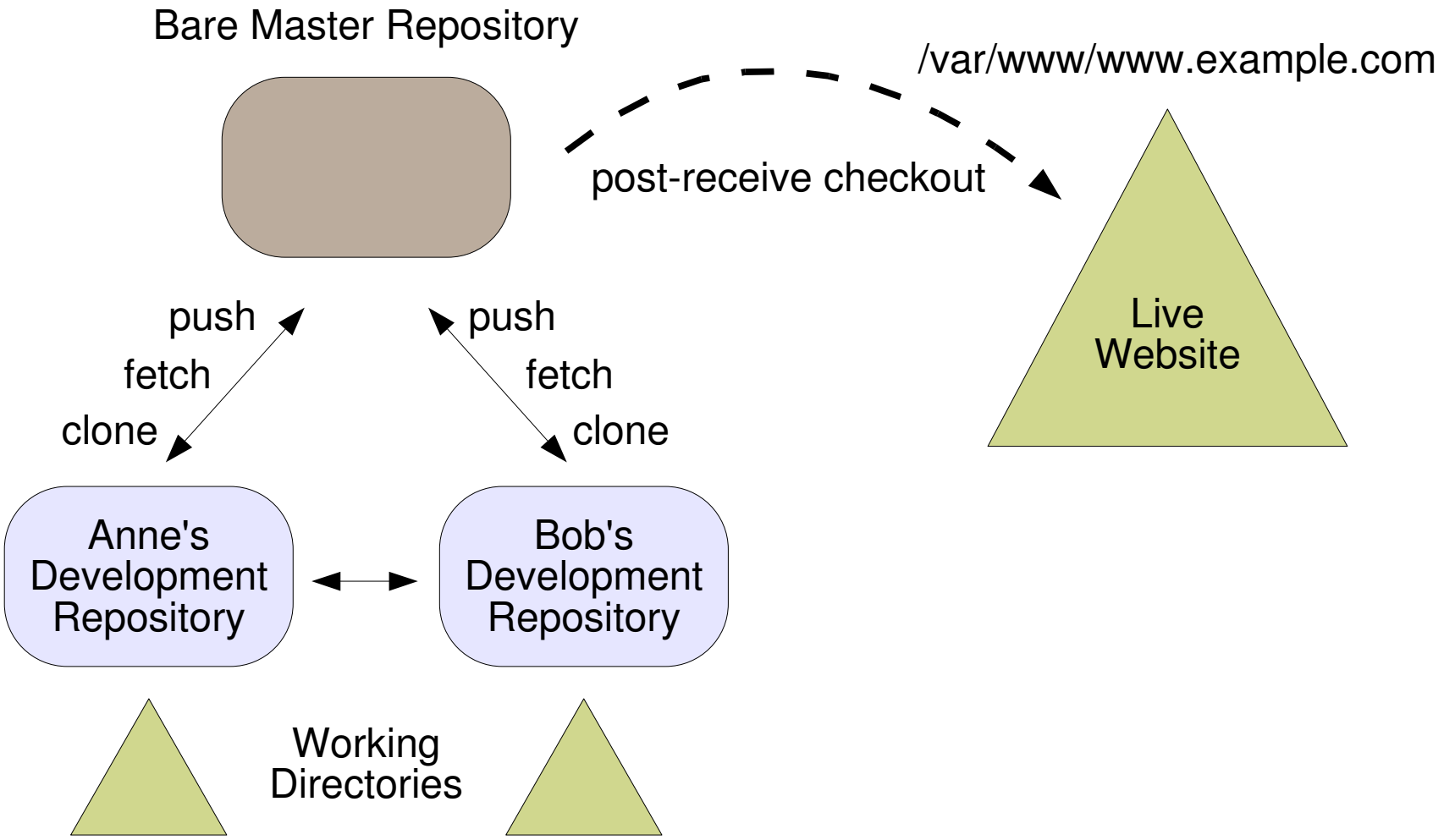
Example: In Place Problems

- Bob is blind-sided
  - Anne's push updates master repository
  - Bob's repository, HEAD mysteriously change
  - Working directory?
- Bob's surprise
  - Use post-receive hook to update working directory?
  - Bob's edits are gone!
- Editing live content!
- Rule: Only push to **bare** repositories.



freescale semiconductor

Example: Website Using Bare Master Repository



freescale semiconductor

Hooks: WebSite Repository Setup

- Make initial bare master repository

```
$ cd /location/of/master/repositories
$ GIT_DIR=WebSite.git git –bare init
$ chmod +x .git/hooks/post-receive          # Wait for it...
```

- Make initial development repository

```
$ cd directory/with/WebSite
$ git init

$ cat index.html
<html>
<body>My website is alive!</body>
<html>

$ git add index.html
$ git commit -m 'Start my website!'
```

Hooks: post-receive Hook

- $ cat WebSite.git/hooks/post-receive # In bare repository

```
#!/bin/sh
WEBROOT_DIR=~jdl/public_html
GIT_WORK_TREE=${GIT_WORK_TREE-$WEBROOT_DIR}

if [ "$GIT_DIR" = "." ]; then
    GIT_DIR=`pwd`
fi

while read oldrev newrev ref ; do
    if [ "$ref" = "refs/heads/master" ]; then
        echo "Updating $GIT_WORK_TREE"
        echo "Using $ref, now at $newrev"
        if [ ! -d "$GIT_WORK_TREE" ]; then
                mkdir "$GIT_WORK_TREE"
        fi
        cd $GIT_WORK_TREE
        git --work-tree=$GIT_WORK_TREE reset --hard $ref
    fi
done
```

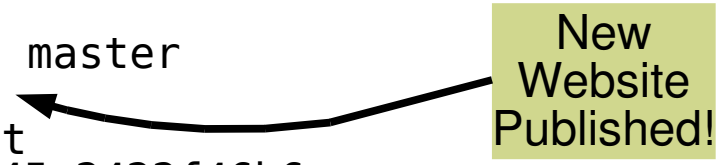
Hooks: Publish New Content

- Commit new content

```
$ git commit -a -m 'New content!'
   Created commit 1bb1c38: New content!
    1 files changed, 2 insertions(+), 1 deletions(-)
```

- Push to publish new website

```
$ git push publish
   Counting objects: 5, done.
   Compressing objects: 100% (2/2), done.
   Unpacking objects: 100% (3/3), done.
   Writing objects: 100% (3/3), 306 bytes, done.
   Total 3 (delta 0), reused 0 (delta 0)
   To /home/jdl/SW/WebSite.git
      09793d9..1bb1c38  master -> master
   Updating /home/jdl/public_html
   Using refs/heads/master, now at
     1bb1c3840f5ad19c53f95a636c1545a2422f46b6
   HEAD is now at 1bb1c38 New content
```



freescale™ semiconductor

Slick Projects: etckeeper

- Problem:
  - Version control /etc config files!
- Solution:
  - etckeeper from Joey Hess
- Features:
  - Tracks /etc configuration files
  - Tracks file metadata -- critical for /etc
  - Automatically commit package upgrades changes
  - Simple and easy
  - Modularly supports other VCS and package systems
- Source:
  - git clone git://git.kitenet.net/etckeeper.git
  - http://kitenet.net/~joey/code/

Project etckeeper: Package Update

- Install a new package as root

```
# apt-get install wu-ftpd
   The following NEW packages will be installed: wu-ftpd
   Get:1 http://ftp.us.debian.org etch/main wu-ftpd 2.6.2-25 [287kB]

   Created commit fb78ef0: saving uncommitted changes in /etc prior
     to apt run
    2 files changed, 7 insertions(+), 0 deletions(-)

   Preconfiguring packages ...
       [...]
   Setting up wu-ftpd (2.6.2-25) ...

   Created commit 9fc806d: committing changes in /etc after apt run
    22 files changed, 310 insertions(+), 0 deletions(-)
    create mode 100644 ftpusers
    create mode 100755 init.d/wu-ftpd
        [...]
    create mode 100644 wu-ftpd/welcome.msg
```

Project etckeeper: Concerns

- Cautions:
  - /etc contains sensitive files
  - “Development” as root
  - You still have to push to off-site backup
  - You still have to push to a **trusted** off-site backup
  - Live working /etc directory – checkout with caution!
- Questions:
  - Branch for original Debian version and yours?
  - Hmm. Branch for separate hosts?

Slick Projects: gitosis

- Problem:
  - Access control to hosted Git repositories
- Solution:
  - Gitosis from Tommi Virtanen
- Features:
  - Gitosis provides easy Git repositories hosting
  - Multiple repositories under one hosting account
  - Uses SSH keys for secure access
  - Shell access on the host is not necessary
  - Provides access control
    - Read/write per user per repository
- Source:
  - git clone git://eagain.net/gitosis.git
  - http://scie.nti.st/2007/11/14/hosting-git-repositories-the-easy-and-secure-way

Project gitosis: New Config Repository

- Install gitosis

```
# Become root on securehost.example.com
# Install gitosis according to README!
```

- Add new hosted repository

```
# cd gitosis-admin
# emacs gitosis.conf
   [group my-host-com-etc]
   members = root@my-host.com
   writable = my-host-com-etc

# emacs keydir/root@my-host.com.pub
   ssh-rsa AAAAB3Nza .... == root@my-host.com

# git add keydir/root@my-host.com.pub
# git commit -m“Add my-host-com-etc repository config”
```

Project gitosis: Secure Repository Backup

- Setup offsite Backup Repository

```
# git remote add backup
      git@securehost.example.com:my-host-com-etc.git
```

- Push to it

```
# git push backup master:refs/heads/master
   Pushing to git@securehost.example.com:my-host-com-etc.git
   Enter passphrase for key '/root/.ssh/id_rsa':
   Initialized empty Git repository in ...../my-host-com-etc.git/
   Counting objects: 1424, done.
   Compressing objects: 100% (971/971), done.
   Writing objects: 100% (1424/1424), 3.35 MiB | 3431 KiB/s, done.
   Total 1424 (delta 173), reused 1229 (delta 97)
   To git@securehost.example.com:my-host-com-etc.git
    * [new branch]      master -> master
   updating local tracking ref 'refs/remotes/backup/master'
```

Slick Projects: ikiwiki

- A Wiki by Joey Hess
  - Wiki, Blog, Bug Tracking System, etc.
  - Full version control for entire Wiki site
  - Plugin modules
- Version control using a Git back-end
  - Supports web-based page editing
  - Supports editing cloned repositories
- Uses multiple Git repository model
  - Bare master repository
  - Web-based edits occur in a cloned repository
  - Personal development occurs in a cloned repository
  - Live site updates via post-commit hook from master repository

Future Git Directions

- Native port to a pacific north-west operating system
  - Well under way – projected for 1.6.0
- Slow progress towards libifying Git
  - And the continuing conversion to C
- Submodule support
  - Present but evolving still...
- Miscellaneous
  - Improve user-interface
  - Simplify command structure
  - Patch sequence editor
  - Case insensitive file system support?
  - Better hunk editing?

Questions?

- Questions?


freescale™
semiconductor