Using Git Branches with Your Team

Craig Golightly

Senior Software Consultant

@seethatgo | www.seethatgo.com

Overview

Remotes, pull, push

Pull requests

Avoiding conflicts

- `.gitignore` file

Best practices for team members

What can go wrong?

- Whitespace errors
- Break the build

Remotes

Location where project is hosted

Often the central repository but not always

- GitHub
- Bitbucket
- GitLab
- Anywhere

Reference for project

- Team milestones
- Testing
- Code reviews

GitHub Fork

Creates a copy of a repo in your account

Use forked copy as remote for changes

Can follow along with demo by forking example

Demo

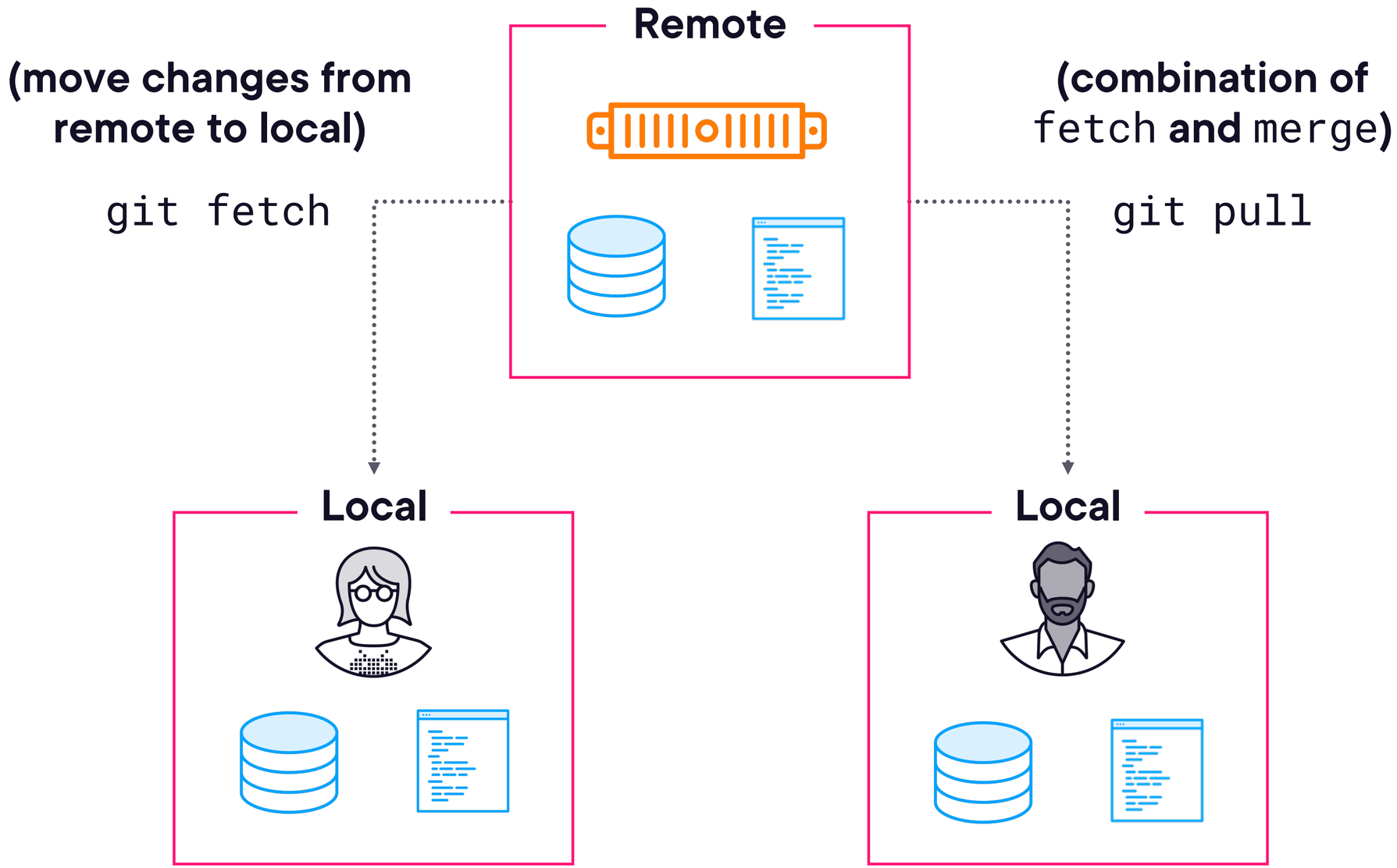
Existing GitHub repository

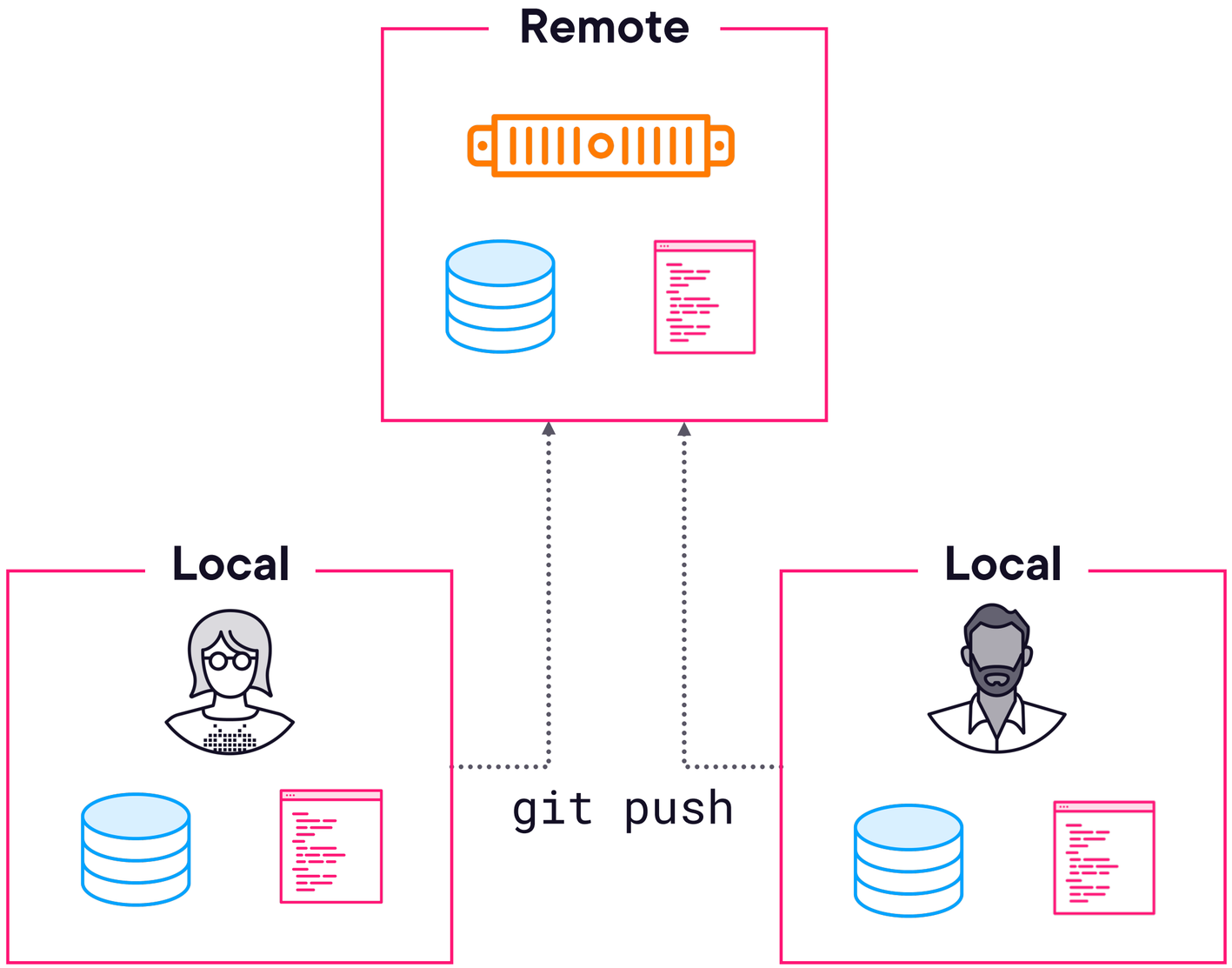
Clone repository to local machine

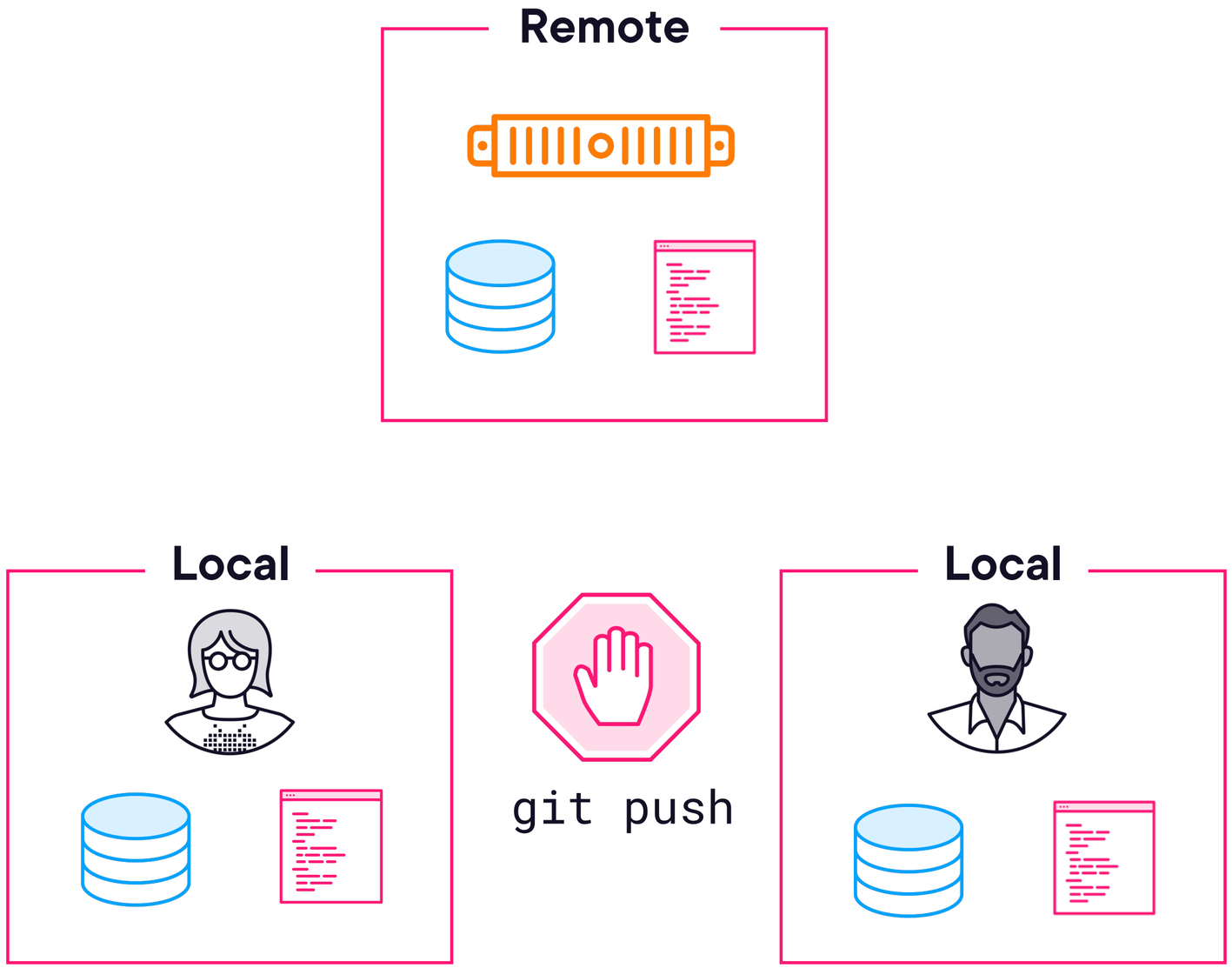
Verify remote setup

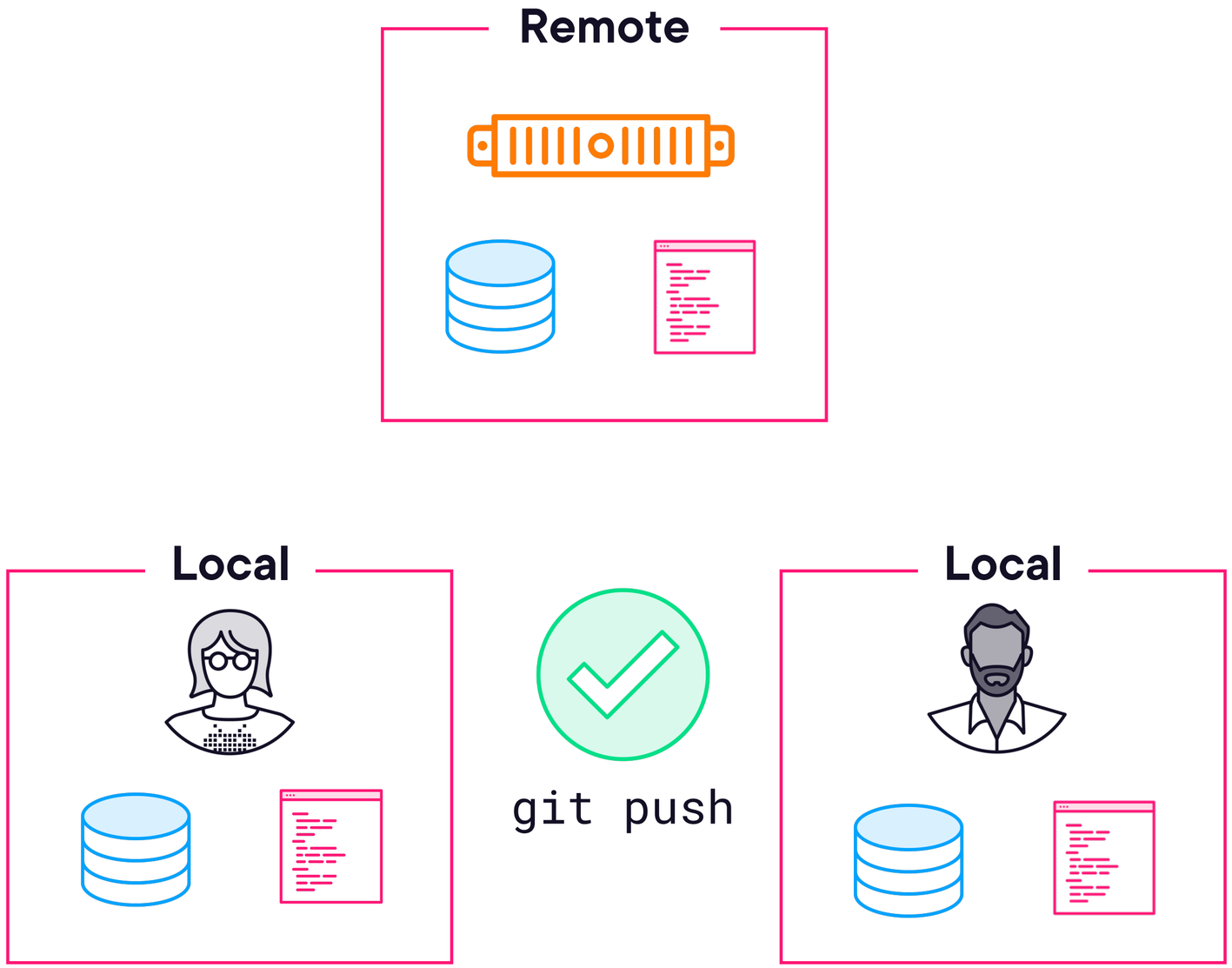
```
https://service.com/project/a.git

git clone <remote-url>

origin

git remote add <name> <remote-url>

git remote -v
```

- **Remote URL**
- **Pull code and set up local branch**
- **Default name for remote server**
- **Provide the name for a remote server**
- **List remotes and URLs**


Remote
(move changes from remote to local)
git fetch
(combination of fetch and merge)
git pull
Local
Local


Remote
Local
Local
git push


Remote
Local
Local
git push

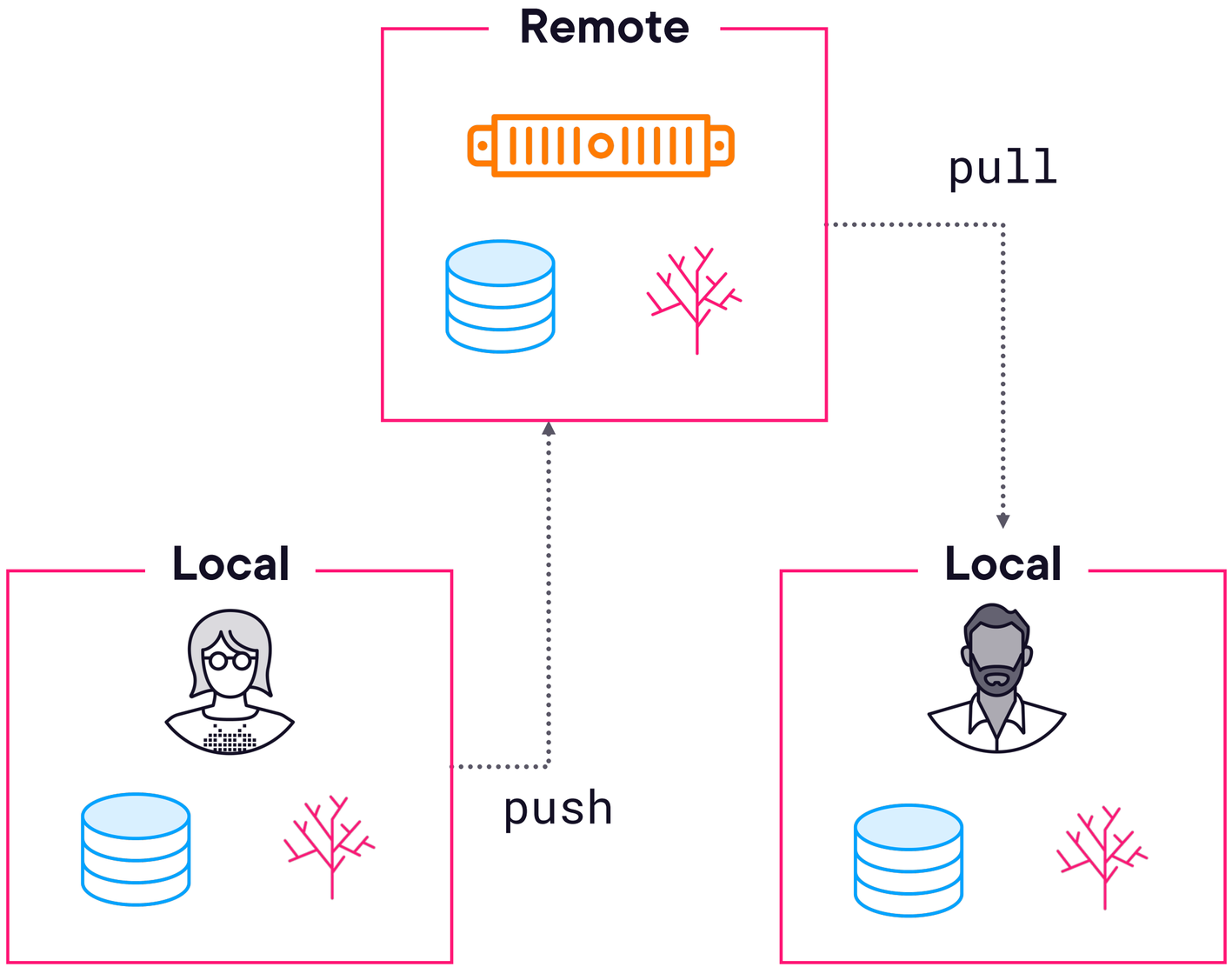

Remote
Local
git push
Local

Demo

Pull a change from remote to local

Push a change from local to remote

Managing concurrent changes to repo


Remote
pull
Local
push
Local

Demo

Create a local branch with changes

Push the branch to remote

List remote branches

Fetch and checkout branch

```
git push -u origin feature4
```

- Push the feature4 branch to remote and set origin as the upstream branch

```
git ls-remote
```

- List branches on remote (no space)

```
git remote -v
```

- List remotes and URLs

```
git fetch origin feature4
```

- Fetch the feature4 branch from remote to local

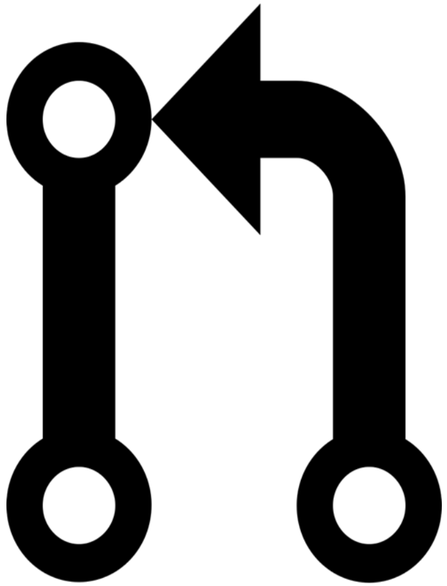
```
git branch -a
```

- List both remote-tracking and local branches

```
git checkout --track origin/feature4
```

- Set up a local branch to track the remote branch

Pull Request



Feature of GitHub (company)

- Not a part of Git (technology)

PR

- "I sent you a PR to review"
- "I opened a PR with the changes"

Finished work in a branch

Facilitates review and testing

Request to merge code into main branch

Pull Request Workflow

Push branch to remote
Open pull request

Add reviewers
Discuss changes

Make fixes in local branch
Push to remote

Do NOT rebase
After pushing to remote

Merge to main
Delete branch

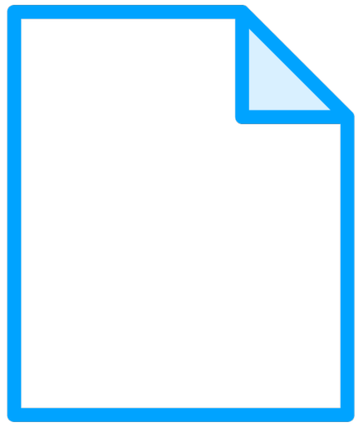
Demo

Feature branch for new code

Open a pull request for feedback

Merge and delete feature branch

Ignore File



`.gitignore`

Files are

- Tracked
- Untracked
- Ignored

Generated files are usually ignored

- Most likely to conflict

Ignored files not added to branches

```
* ? ! / [a-zA-Z]

# this is a comment

**/bin

*.zip

/bin

git rm --cached <filename>
git rm <filename>

.git/info/exclude
```

- anything, one character, negator, directory separator, range
- # for comments

- ** matches any directory in repository

- * matches any file in repository

- relative to .gitignore directory
  Put a .gitignore at your project root!

- Delete file from Git repo or
  delete from repo and local filesystem

- Ignore patterns for your system only

Demo

Create file ignored by `.gitignore`

Customize ignore rules for your workspace

Find example `.gitignore` files

Prevention and Treatment

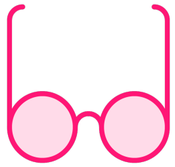
Code conflicts

Merging tools

Team conventions and best practices

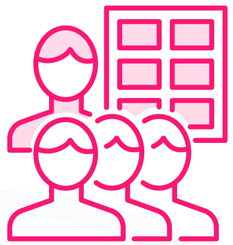
Team Conventions

`.gitignore` **file**



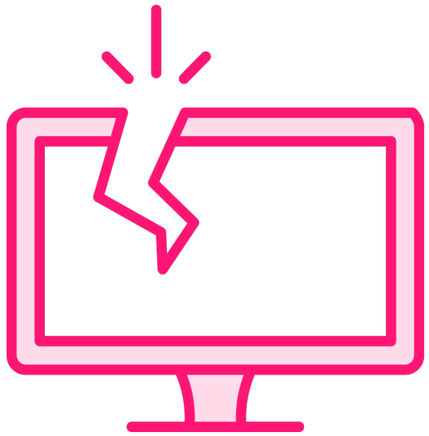
`README.md`

Frequent smaller commits vs. one giant commit



Coordination of large refactors, renames, reformats

Don't Break the Build



Automated Tests

Timer on main

Run on branch

Errors

Syntax errors

Unit test failures

"On it."

Get help

Report when fixed

Merge Tips

Update from remote

Before attempting a merge

Avoid Errors

Build code on local machine

Run unit tests locally

Fix any errors or failures

Commits

Commit related changes

Would commit make sense on its own?

Good subject line

What commit does

< 50 characters

Add a body (when appropriate)

WHY change was made

```
//int function foo() {
//  int x = 1;
//  int y = 2;
//  return x + y;
//}
```

Commented Out Code

If it was committed, Git will remember it

Costs time of everyone who reads it

Let Git do its job of keeping a history

535 src/main/java/rocks/nt/apm/jmeter/JMeterInfluxDBBackendListenerClient.java

```
@@ -1,286 +1,277 @@
- package rocks.nt.apm.jmeter;
-
- import java.util.ArrayList;
- import java.util.HashSet;
- import java.util.List;
- import java.util.Set;
- import java.util.Random;
- import java.util.concurrent.Executors;
- import java.util.concurrent.ScheduledExecutorService;
- import java.util.concurrent.TimeUnit;
-
- import org.apache.jmeter.config.Arguments;
- import org.apache.jmeter.samplers.SampleResult;
- import org.apache.jmeter.threads.JMeterContextService;
- import org.apache.jmeter.threads.JMeterContextService.ThreadCounts;
- import org.apache.jmeter.visualizers.backend.AbstractBackendListenerClient;
- import org.apache.jmeter.visualizers.backend.BackendListenerContext;
- import org.apache.jorphan.logging.LoggingManager;
- import org.apache.log.Logger;
- import org.influxdb.InfluxDB;
- import org.influxdb.InfluxDBFactory;
- import org.influxdb.dto.Point;
- import org.influxdb.dto.Point.Builder;
-
- import rocks.nt.apm.jmeter.config.influxdb.InfluxDBConfig;
- import rocks.nt.apm.jmeter.config.influxdb.RequestMeasurement;
- import rocks.nt.apm.jmeter.config.influxdb.TestStartEndMeasurement;
- import rocks.nt.apm.jmeter.config.influxdb.VirtualUsersMeasurement;
+ public class JMeterInfluxDBBackendListenerClientpackage rocks.nt.apm.jmeter;
+
+ import java.util.ArrayList;
+ import java.util.HashSet;
+ import java.util.List;
+ import java.util.Set;
+ import java.util.Random;
+ import java.util.concurrent.Executors;
+ import java.util.concurrent.ScheduledExecutorService;
+ import java.util.concurrent.TimeUnit;
+
+ import org.apache.jmeter.config.Arguments;
+ import org.apache.jmeter.samplers.SampleResult;
+ import org.apache.jmeter.threads.JMeterContextService;
+ import org.apache.jmeter.threads.JMeterContextService.ThreadCounts;
+ import org.apache.jmeter.visualizers.backend.AbstractBackendListenerClient;
+ import org.apache.jmeter.visualizers.backend.BackendListenerContext;
+ import org.apache.jorphan.logging.LoggingManager;
+ import org.apache.log.Logger;
+ import org.influxdb.InfluxDB;
+ import org.influxdb.InfluxDBFactory;
```

Whitespace is invisible, right?

- Not to Git

Can drown out real changes in code

Pick a formatting standard

- Can automate it

Line endings

- Differences can cause problems
- Pick one and enforce it

Summary

Use and manage remotes

Pull request

- Easier code reviews
- Coordinate merges

Avoid breaking the build

- Check locally before merge

Frequent updates and smaller merges

Avoid conflicts

- `.gitignore` file
- Consistent whitespace settings

Up Next:

Advanced Merging Methods