Documentación y optimización

- Diapositivas
- Actividades

Introducción

En esta Unidad aprenderemos a:

- Trabajar de forma habitual con un sistema de control de versiones.
- Identificar los patrones de refactorización más usuales.
- Revisar el código fuente usando un analizador de código.
- Documentar el código fuente.

Optimización

Hediondez del código

Que huele o apesta !!! Que tiene mala pinta!!!

- También llamado **code smell** en inglés
- Es síntoma en el código fuente que indica posiblemente un problema más profundo.
- Usualmente no son bug de programación (errores): no son técnicamente incorrectos y en realidad no impiden que el programa funcione correctamente.
- Indica deficiencias en el diseño que puede ralentizar el desarrollo o aumentan el riesgo de errores o fallos en el futuro.
- Es un motivo importante para realizar refactorización.
- Hediondez del código

Análisis de código

- Tipos:
  - **Análisis dinámico** (unit tests)
  - **Análisis estático** (**lint**)

Análisis estático de código

- Mediante analizadores estáticos (**linters**)
- Introducción a los linters -en inglés-
- Mediante sitios web para inspección de código (**Continuous Inspection**)

Linters

- Analizadores estáticos de código:
- **lint**: C
- **sonar**: Java
- **JSLint**, **ESLint**: Javascript

Continuous Inspection o Continuous Analysis

- Sitios web que ofrecen **inspección de código**:
- **Scrutinizer**
- **SonarQube**

Scrutinizer

- PHP, Python y Ruby soportados
- 14 días de prueba gratis
- Precio/Mes:
  - Plan Basic: 49 €
  - Plan Professional: 99 €
  - Plan Unlimited: 199 €

Note: A fecha Diciembre 2017

SonarQube

- Más de 20 lenguajes soportados
- Precio/Mes:
  - Plan Open Source: 0 $
  - Plan Private Projects: Desde 5 $

Note: A fecha Diciembre 2017

Refactorización

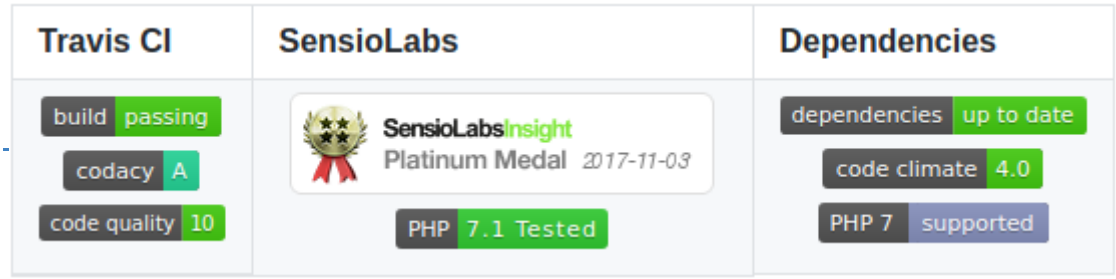
- Es el proceso de reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.
- Técnicas:
  - Renombrado de variables
  - Pasar código duplicado a funciones
  - Eliminación de código inalcanzable
  - Eliminación de código redundante
  - Eliminación de código muerto
  - ...

Artículo en Wikipedia **echadle un vistazo al articulo y a sus descendientes: Véase También**

Documentación

Insignias (badges)

Imágenes que se insertan e indican un estado



Tipos de documentación

- Documentación de código
- Documentación técnica
- Documentación de usuario

Formatos de documentación

- **HTML** (p. ej. Javadoc)
- **Markdown** (p. ej. Gitbook)
- **reStructuredText** (p. ej. Readthedocs)
- **asciiDoc**

Control de versiones

Sistemas más conocidos:

- CVS
- Subversion
- Mercurial
- **Git**

Git

https://www.atlassian.com/es/git/tutorials → con Bitbucket
https://github.com/JJ/aprende-git → con Github

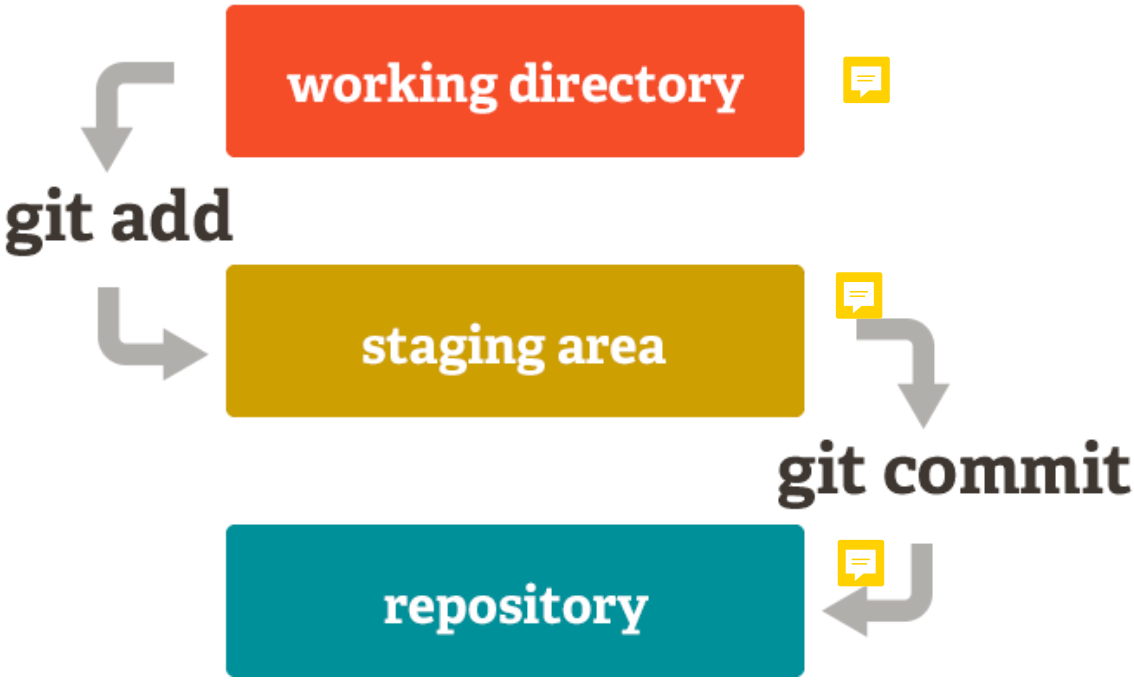
Características

- Moderno
- Distribuido
- Eficiente

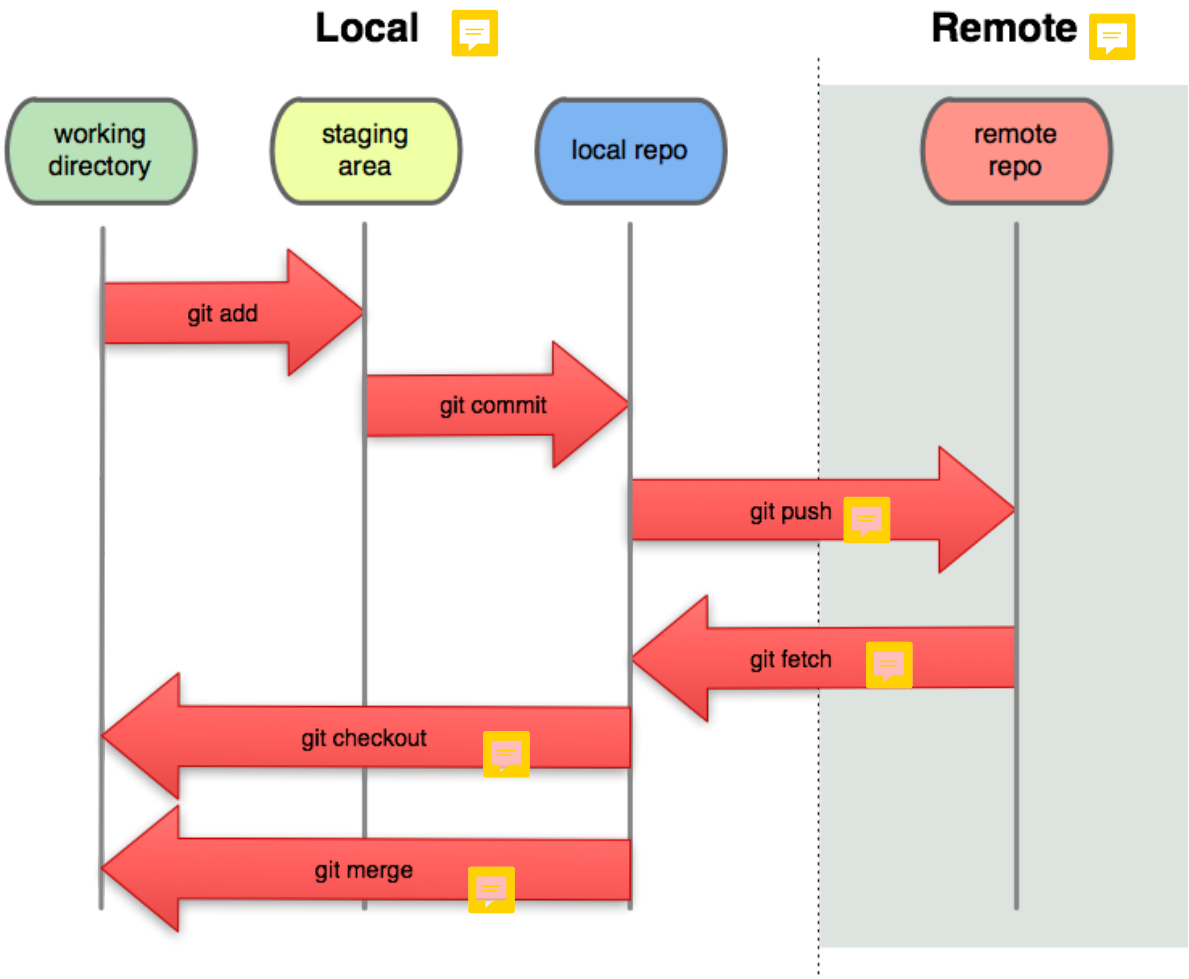
Git (Conceptos)

- Repository (local & remote)
- Commit
- Branch
  - Checkout
  - Merge (fast-forward, 3-way)

Git (Áreas)



Git (Áreas)



Git (Ramas)

Git (Comandos)

```
# Configuración
  config

# Repositorios
  clone, remote add, remote rm

# Básicos
  init, status, log, add, rm, commit, push, pull

# Ramas (branches)
  branch, branch -d, merge, checkout, stash

# Otros
  diff, tag, submodule
```

CheatSheet

Sitios que soportan GIT

- **GitHub**
- **Bitbucket**
- **GitLab**
- **Coding -en chino-**