UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

Facultad de Ingeniería de Sistemas Computacionales

Kexy Rodríguez
kexy.rodriguez@utp.ac.pa

Buenas Prácticas de Programación

Agosto de 2020

Contenido

- Pilares de las buenas prácticas de programación
- Organización
- Estructuración
- Orden
- Estilos de escrituras
- Documentación
- Control

Pilares de las buenas prácticas de programación

1.

Organización

- Estructuración

2.

Orden

- Escritura
- Documentación

3.

Control

- Control de versiones

1.

Organización

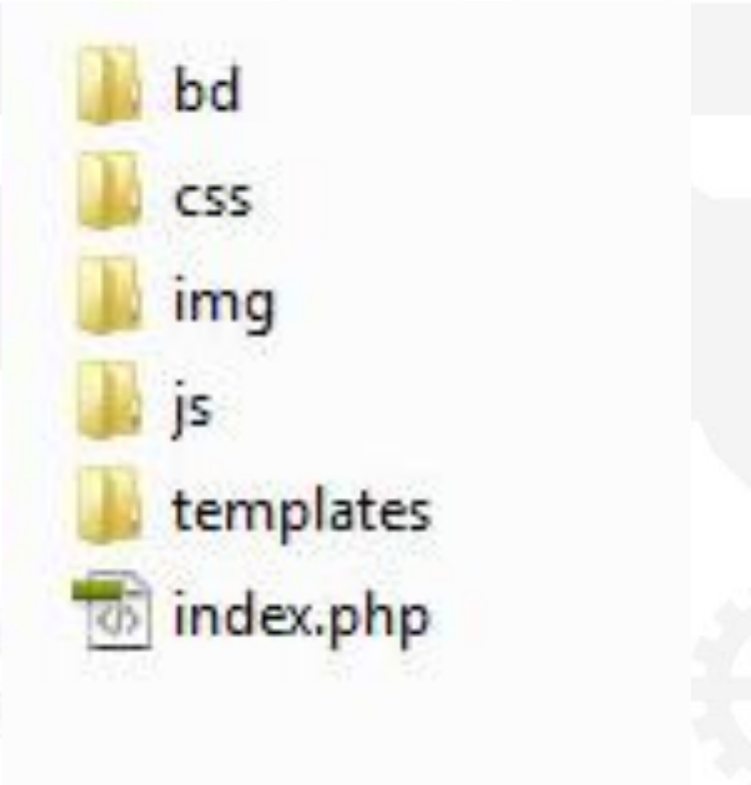
- Estructuración

Organización

- **Estructuras de los directorios del proyecto**

  Es importante tener un orden en los archivos que se generan en el desarrollo de una aplicación.

  Como recomendación los archivos pueden ser agrupados en carpetas según tipo o función.



(Marchena, 2017)

Organización

- Existen diferentes **formas de organizar nuestros directorios** del proyecto.
- Hoy en día tenemos la ventaja que la organización puede ya estar definida por el ***framework*** o **IDE** que utilicemos:

Los framework o IDE nos ayudan a tener nuestros archivo de manera organizada; sin embargo, es importante llevar una organización sin importar la utilización de una herramienta o no.

(Marchena, 2017)

Organización

Indentación

La indentación es un tipo de notación secundaria utilizado para **mejorar la legibilidad del código** fuente por parte de los programadores.

En ciertos lenguajes de programación como Haskell, Occam y **Python**, se utiliza para delimitar la estructura del programa permitiendo establecer bloques de código.

```
public void conIndentacion(){
    int i, j;
    for (i = 0; i <= 10; i++){
        for (j = 0; j <= 10; j++){
        System.out.print("%i x %i = %i\n"+ i + j + i * j);
        }
    }
}

public void sinIndentacion(){
int i, j;
for (i = 0; i <= 10; i++){
for (j = 0; j <= 10; j++){
System.out.print("%i x %i = %i\n"+ i + j + i * j);
}
}
}
```

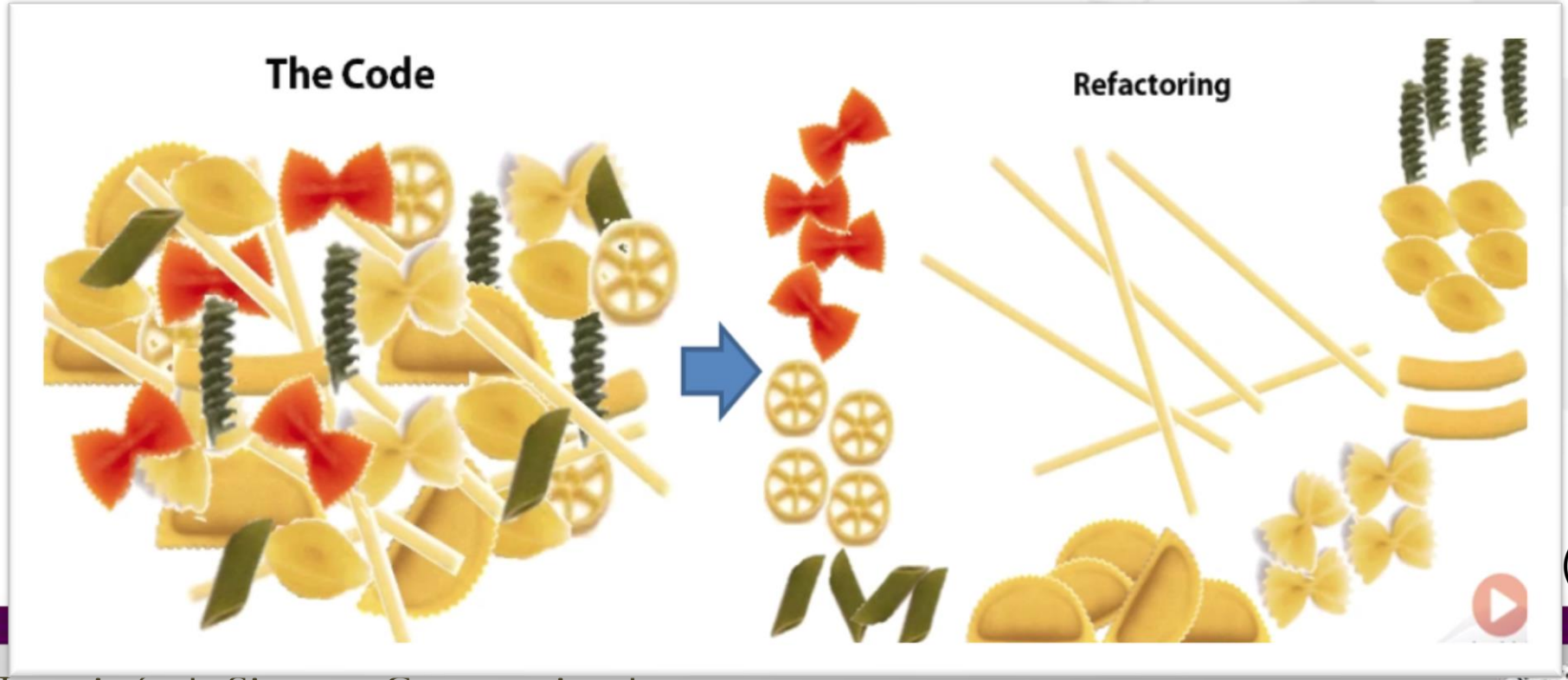
Con Indentación

```
nota = float(input("Introduce una nota: "))
if nota >= 9:
    print("Sobresaliente")
if nota >= 7 and nota < 9:
    print("Notable")
if nota >= 6 and nota < 7:
    print("Bien")
if nota >= 5 and nota < 6:
    print("Suficiente")
if nota < 5:
    print("Insuficiente")
```

Organización

Refactorización

Es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo. La refactorización es la parte del mantenimiento del código que no arregla errores ni añade funcionalidad.



(Marchena, 2017)

2.

Orden

- Escritura
- Documentación

Estilos de Escritura

camelCase

SNAKE_CASE

PascalCase

Escritura

Variables

- **camelCase**: se utiliza para nombrar **variables de trabajo**. Se coloca la primera letra en minúscula y la siguientes palabras la primera letra en mayúscula. Ejemplo: **s**egundo**N**ombre

```php
if($val==1) {
    $tmpName = "";
    $nombreFile = "";
    $directorio = "informe";
```

Escritura

Constantes

- **SNAKE_CASE**: utilizada para nombrar **constantes**. Se coloca todo en mayúscula cerrada separado por guion abajo (_).
- Ejemplo: SEGURO_EDUCATIVO=0.125

```
=> 'pgsql',
=> env('DB_HOST', 'localhost'),
=> env('DB_DATABASE', 'forge'),
=> env('DB_USERNAME', 'forge'),
=> env('DB_PASSWORD', ''),
=> 'utf8',
```

Escritura

Clases, los módulos y funciones

- **PascalCase**: se utiliza esta notación para nombrar las **clases, los módulos y funciones**.
- Todas las primeras letras van en mayúscula, ayuda a que se haga la interpretación de que estamos en un archivo o función importante.

```
def CalcularSuma(numero1,numero2):
    print numero1 + numero2
    print "\n"
```

Escritura

Otras reglas básicas de escritura de código son:

- ✓ Los **nombres de las funciones** deben ser **verbos y empezar en mayúscula**.
- ✓ El **nombre del objeto es el mismo que la clase** aplicando camelCase.
- ✓ Si el lenguaje lo permite, **separa los valores de los operadores**.
- ✓ Recuerda siempre **inicializar la variable**.

Documentación

- Para reducir la cantidad de comentarios, podemos **utilizar nombres descriptivos en las variables**.
- Utilice oraciones completas al escribir comentarios. **Los comentarios deben aclarar el código, no añadir ambigüedad**.
- Utilice los comentarios para **explicar la intención del código**. No deben servir como traducciones en línea del código.
- **Al comienzo** de cada rutina, es útil **proporcionar comentarios que indiquen el propósito**, las suposiciones y las limitaciones de la rutina
- Comente el código en el momento, porque lo más probable es que no habrá tiempo para hacerlo más tarde. Además, si tiene la oportunidad de revisar el código que ha escrito, lo que es obvio hoy probablemente no será obvio seis semanas a partir de ahora.

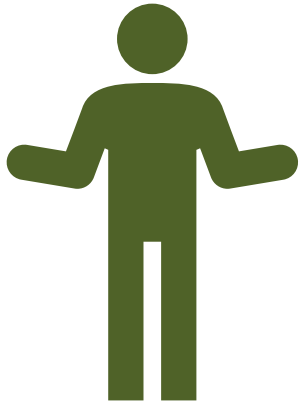
3.

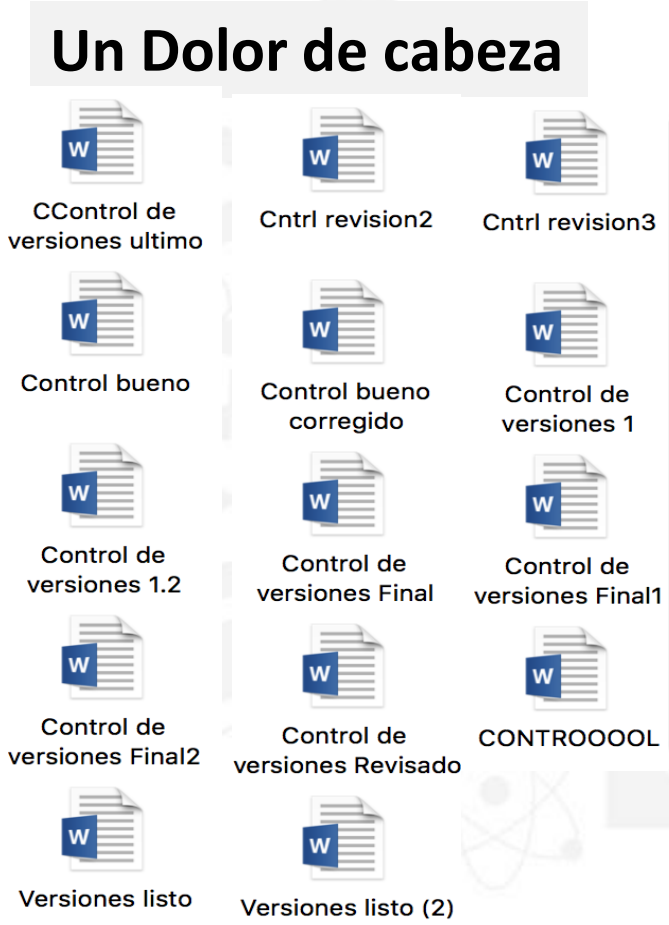
Control

- Control de versiones

Control

Muchas veces cuando estamos trabajando en un proyecto, creamos muchos archivos con el mismo contenido. Esto genera confusión al momento de tener un control de la última versión del documento, archivo o código de cualquier proyecto.





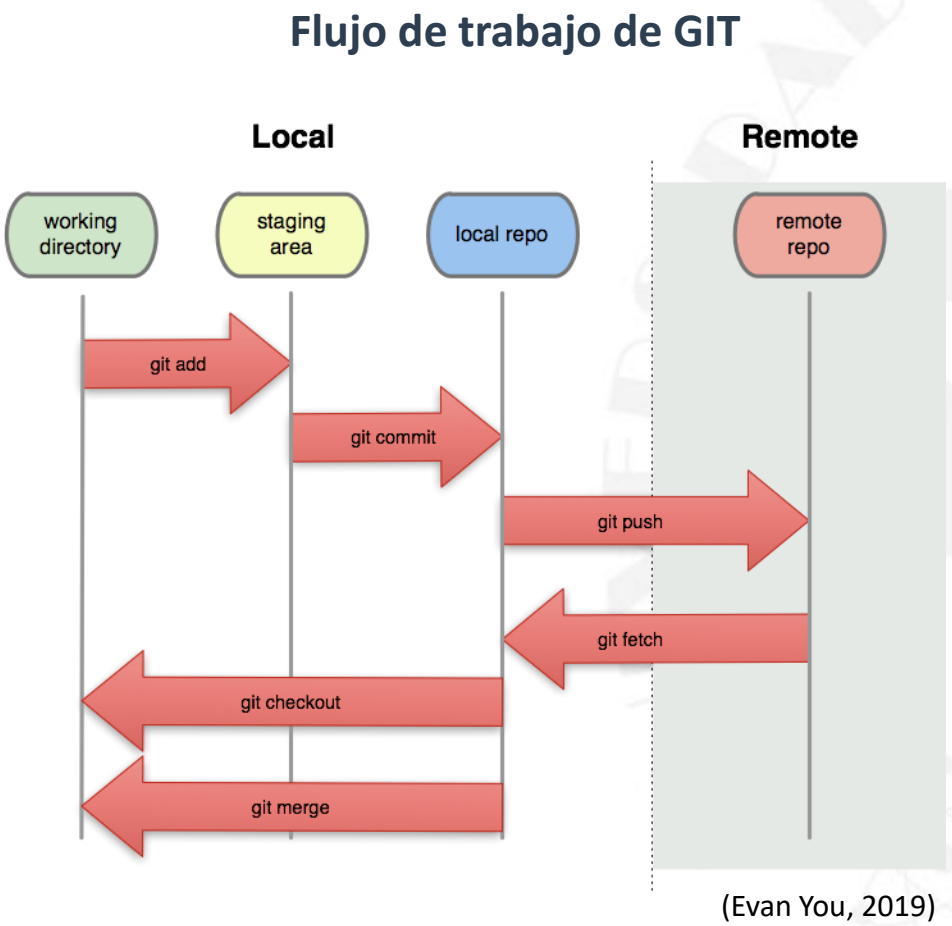
Control

Actualmente existen herramientas basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de archivos. Estos son llamados control de versiones.

“Un control de versiones es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante. Aunque en los ejemplos de este libro usarás archivos de código fuente como aquellos cuya versión está siendo controlada, en realidad puedes hacer lo mismo con casi cualquier tipo de archivo que encuentres en una computadora.”

Una Solución

Control



(Evan You, 2019)

https://rogerdudler.github.io/git-guide/index.es.html
https://bluuweb.github.io/tutorial-github/guia/

Control

¿Por qué debería ser importante el control de versiones?

- Revertir archivos a un estado anterior, revertir el proyecto entero a un estado anterior
- Te permite, **comparar** cambios a lo largo del tiempo, **ver quién modificó** por última vez algo que puede estar causando un problema, **quién introdujo un error y cuándo**, y mucho más.

Viajar al Pasado

Resumen

- ¿Por qué es importante las Buenas prácticas de programación?
- ¿Qué es y qué diferencia existe entre un framewok, IDE y un CMS?
- ¿Cuáles son los tres pilares de las Buenas prácticas de programación?
- ¿Cuáles son los tres estilos de escrituras visto en esta presentación y para qué se utilizan cada uno?
- ¿Por qué es importante la indentación?

¿Cómo se transmite el COVID-19?

Por contacto personal cercano con una persona infectada.

Según la OMS, sobre la base de datos actuales a nivel global, 81% de los pacientes con COVID-19 presentan un cuadro leve del virus.

A través de personas infectadas al **toser o estornudar.**

Al tocar objetos o superficies contaminadas y luego tocarse la boca, la nariz o los ojos.

REPÚBLICA DE PANAMÁ
GOBIERNO NACIONAL

MINISTERIO DE SALUD

OPS
Organización Panamericana de la Salud
Organización Mundial de la Salud

C S S
PANAMÁ