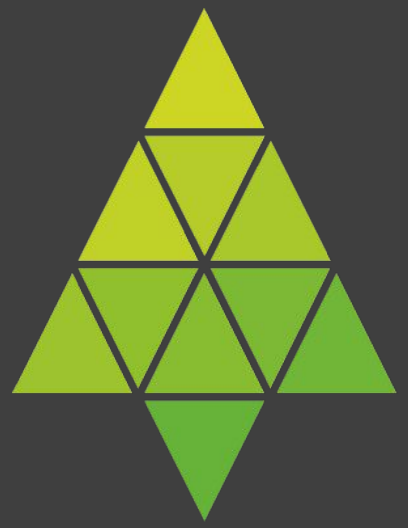


10 Pines

Diseño a la Gorra - Episodio 07

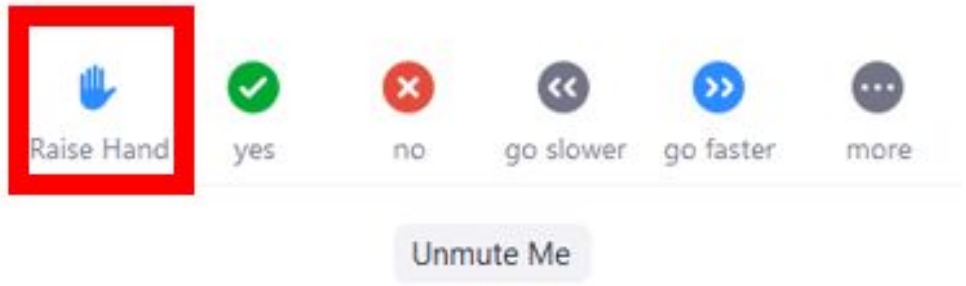
Hernán Wilkinson

hernan.wilkinson@10pines.com

@HernanWilkinson

https://alagorra.10pines.com

Estar muteados a menos que sea necesario



No voy a poder leer el chat

La comunicación visual es importante. Usarla a discreción

Diseño ¡a la gorra!

¡Bienvenidos!
Durante esta serie de Webinars exploraremos *qué* significa **Diseñar Software con Objetos** y *cómo* lo podemos hacer cada vez mejor.
Trataremos muchos temas que irán desde cuestiones filosóficas como qué significa Diseñar en nuestra profesión y dónde está expresado ese Diseño, pasando por consejos y heurísticas para diseñar "mejor" y terminado con ejemplos concretos de cómo aplicar esas heurísticas en la *vida real.*
Los webinars son "*language agnostic*", o sea que no dependen de un lenguaje de programación en particular, aunque los ejemplos que usaremos estarán hechos principalmente en **Java**, **JavaScript**, **Ruby**, **Python** y mi querido **Smalltalk** cuando amerite 😉 .

Te esperamos todos los Martes a las 19 Hrs GMT-3 a partir del Martes 11 de Agosto de 2020.
Para poder participar tenes que registrate acá.

Todo el código y presentaciones estarán disponibles para que lo puedan usar y consultar en cualquier momento acá.

¡Trae ganas de aprender y pasarla bien!

¿Por qué a la Gorra?
Al igual que cuando Diseñamos Software está bueno usar una **Metáfora** para entender qué estamos modelando, en este caso usamos una metáfora para explicar cómo *financiaremos*

Donaciones

mercado pago

$100 - Casi una 🍷 Pagar

$250 - Una buena 🍺 Pagar

$500 - Menos que 🍔 +🍟 Pagar

PayPal Donate

¿Querés donar un monto variado o en otra plataforma?

Dictado por:
Hernán Wilkinson

https://alagorra.10pines.com

Online

Diseño Avanzado de Software con Objetos I

Empieza el **13/10**

Del 13/10 al 22/10. Días: 6 días — Del 13 al 16, 20 y 22 de Octubre — 9:00 a 13:00 GMT-3.

AR$ **15.300-** (IVA incluído)

Hacer una consulta

Inscribirme ahora 15% Dto.

Dictado por:
Máximo Prieto

~~AR$ 18.000-~~

AR$ **15.300-**

(IVA incluído)

~~U$D 300-~~

U$D255-

(impuestos incluídos)

Super Early Bird

~~Early Bird~~

https://academia.10pines.com/courses/96-diseno-avanzado-de-software-con-objetos-i

Online

Test Driven Development Avanzado

Empieza el **02/11**

AR$ **12.750-** (IVA incluído)

Hacer una consulta

Inscribirme ahora 15% Dto.

Del 02/11 al 06/11. Días: Lunes a Viernes — 9:00 a 13:00 GMT-3.

Dictado por:
Hernán Wilkinson

~~AR$ 15.000-~~

AR$ **12.750-**

(IVA incluído)

~~U$D 250-~~

U$D215-

(impuestos incluídos)

Super Early Bird

~~Early Bird~~

https://academia.10pines.com/courses/95-test-driven-development-avanzado


SEMINARIO WEB
El Balance de las
Prácticas Técnicas
y Humanas
en el Agilismo
Fecha: 16 de octubre
ONLINE
GRATUITO
¡y en castellano!
Agile Alliance
Hola, soy HernánWilkinson
Y hablaré sobre: Cómo lograr seguridad
psicológica con nuestros diseños

NERDEAR.LA

Día 4 – Container A

23 de Octubre

17:25 - 17:55

¿Qué es el Diseño de Software y cómo podemos hacerlo bien?

By **Hernan Wilkinson** — Socio en 10Pines SRL

NERDEAR.LA

Día 5 – Container A

24 de Octubre

11:00 - 12:00

Keynote - To be announced

By **To be announced**

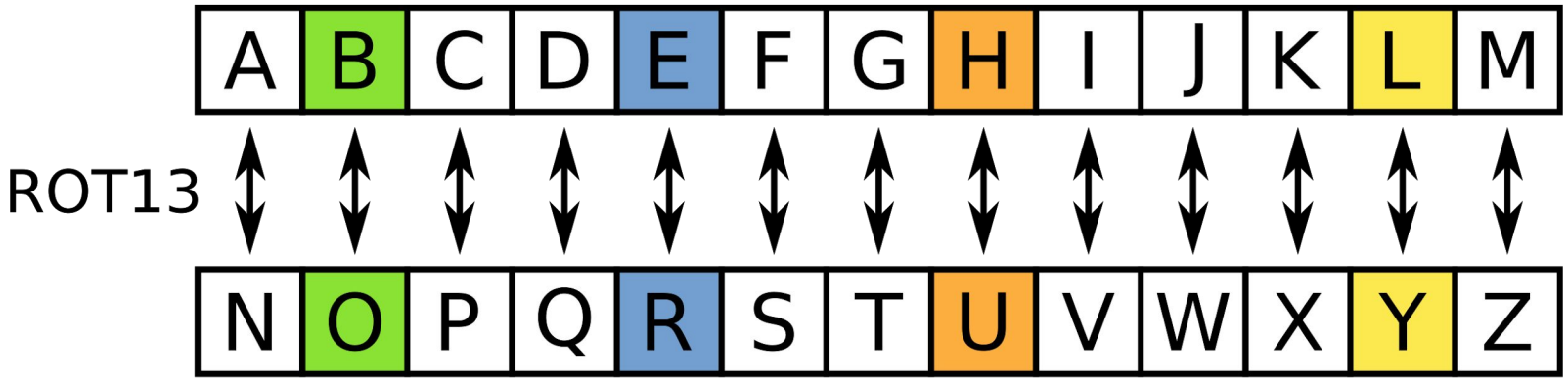
¿Qué vimos en el último episodio?

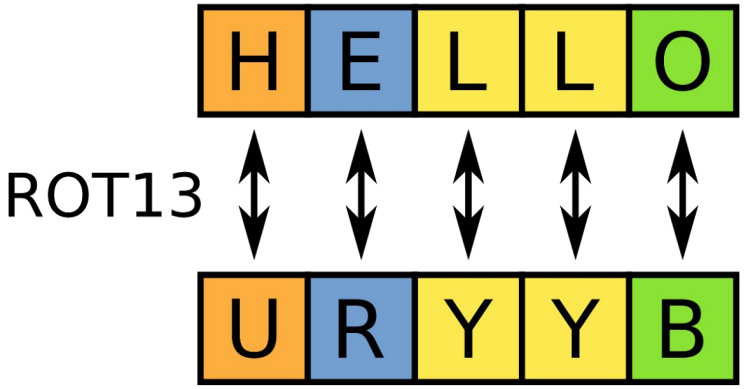
TDD es mucho más que tests que se escriben primero y que hay que hacer pasar después

¿Qué es TDD?

- Técnica de desarrollo basada en características del Aprendizaje
  - Iterativa e Incremental
  - Basada en Feedback Inmediato
- Side-effect:
  - Recuerda todo lo aprendido
  - Y permite asegurarnos de no haber "desaprendido"
- Incluye análisis, diseño, programación y testing

Rot 13





Algunas conclusiones sobre TDD

El tiempo que tardo en cada paso de TDD, es un indicio de qué tan bien estoy realizando la técnica

Los test unitarios están “acoplados” al diseño

Puedo verificar qué tan bien hice los test usando mutation testing

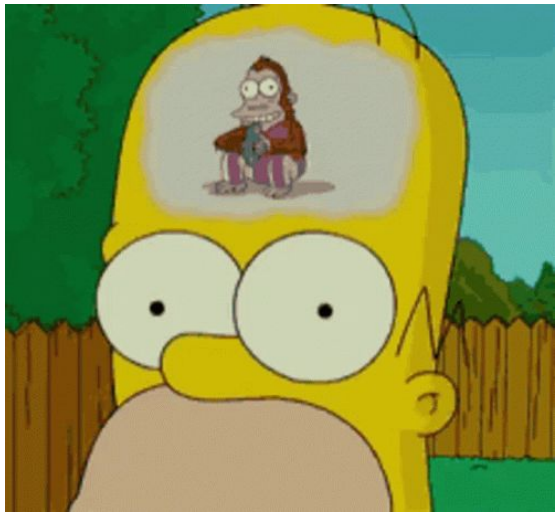
(por lo menos de manera manual)

TDD hace que los programadores sean
los primeros usuarios
del sistema que desarrollan

Un programador no es buen programador si no sabe testear

TDD jerarquizó el testing

TDD hace explícito todos los "tests" que corremos en nuestra cabeza



TDD nos da seguridad al momento de refactorizar

→

Evita que el sistema se convierta en un "sistema legacy"

TDD no implica buen diseño

Ahora si...

Tipos de Tests

Definiciones

- Las definiciones varían según el autor
- Hay definiciones difusas, contradictorias, ambiguas, etc.
- El problema es que hay distintas maneras de categorizar los tests y generalmente se piensa en una sola

Categorías

- De acuerdo al alcance a nivel “código”
- De acuerdo a qué testean
- De acuerdo a cómo están implementados
- De acuerdo a cómo ejecutan
- De acuerdo a cómo se comportan
- De acuerdo a quién lo hace
- De acuerdo a cómo se hacen

De acuerdo al alcance a nivel “código”

- Tests Unitarios (Tests Solitarios)
- Tests de Componentes
- Tests End-To-End/Tests de Sistema/Tests de Integración (Test Sociales)

Tests Unitarios

- Def 1: Verifican el comportamiento de una “unidad”
  - ¿Qué es una unidad? ¿Un método? ¿Una clase? ¿Un “componente”?
- Def 2: Tests que ejecutan en menos de 10 ms
  - Relacionada con la funcionalidad y no con el alcance de código

- Últimamente se los implementa simulando todo lo que la “unidad” bajo test utiliza, por eso también se los llama Test Solitarios

Tests Unitarios

- Ventajas:
  - Cambios en dependencias no impactan en el resultado del tests
  - “Chicos”
- Desventajas:
  - Cambios en dependencias no impactan en el resultado del tests
  - Están acoplados al diseño, si el diseño cambia los tests se ven impactados
  - Gran confusión con TDD debido al nombre del framework xUnit
  - Obligan a “simular” todo aquellos que no es “la unidad testeada”
  - Más orientados a la “implementación” que a la funcionalidad

Tests de Componentes

- Verifican el correcto funcionamiento de un “componente”
  - ¿Qué es un componente? ¿Una clase? ¿Un módulo? ¿Un “java bean”?
- Mismas ventajas y desventajas que los Test Unitarios

Tests de Integración

- Verifican que “partes” implementadas independientemente funcionen “conectadas”
- También llamados *“System test”* o *“End to End tests”* o *“Test Sociales”*

De acuerdo al alcance a nivel “código” - Conclusión

- Según mi opinión, la categoría menos interesante porque tiene que ver con el cómo y no con el qué

De acuerdo a qué testean

- Tests Funcionales
- Tests No Funcionales
  - Tests de Performance
  - Tests de Escalabilidad
  - Tests de Seguridad
  - Tests de Usabilidad

Tests Funcionales

- Verifican una “funcionalidad” sin importar el alcance a nivel “código” (puede abarcar varios métodos, clases, etc)
- Ventajas:
  - Pensados desde el punto de vista funcional y no implementativo
  - Más relacionados con la “reglas de negocio”
- Desventajas:
  - Cambios en alguna dependencia puede hacer fallar varios tests
- Pueden ser unitarios, de integración, etc.

De acuerdo a cómo están implementados

- Tests de Caja Negra
  - Tests que “desconocen” la implementación de lo que testean
- Tests de Caja Blanca
  - Tests acoplados a la implementación de lo que testean

- Paradoja: Cuando hacemos TDD debemos escribir tests de Caja Negra, sin embargo conocemos la implementación

De acuerdo a cómo ejecutan

- Tests Aislados
  - No hay dependencia entre los tests
  - No comparten “datos de prueba”
  - Permiten la paralelización de la ejecución
- Tests Secuenciales
  - Deben ser ejecutados en cierta secuencia
  - El resultado de un test impacta en la ejecución de otro
- Tests Compartidos
  - Test con “datos de prueba” compartidos, ej. la misma base de datos

De acuerdo a cómo se comportan

- Tests Determinísticos
  - Siempre dan el mismo resultado
- Tests Erráticos
  - A veces funcionan, a veces no funcionan
  - No cumplen la regla de “El test debe estar en control de todo”
  - Muy común que suceda con Tests Compartidos
- Tests Frágiles
  - Fallan al cambiar la implementación de lo que se testea, no al cambiar el qué de lo que se testea
  - Ejemplo: Tests con mocks, tests de UI
- Tests “Insoportables”
  - Tests que tardan mucho :-)

De acuerdo a quién lo hace

- Test de Programador
  - Son los tests escritos por los programadores y tienen por objetivo “ayudar” al programador en su proceso de desarrollo
  - Pueden ser funcionales, no funcionales, unitarios, de integración, desarrollados haciendo TDD o Testing, etc.
- Test de QA
  - Son los tests creados por gente que no tiene por objetivo principal programar

De acuerdo a cómo se hacen

- Tests Automatizados
  - Tests programados
  - Tests record & play
- Tests Manuales
  - Pre-definidos
  - Exploratorios

TDD: Tipos de Tests y sus Características

Tipo de Tests

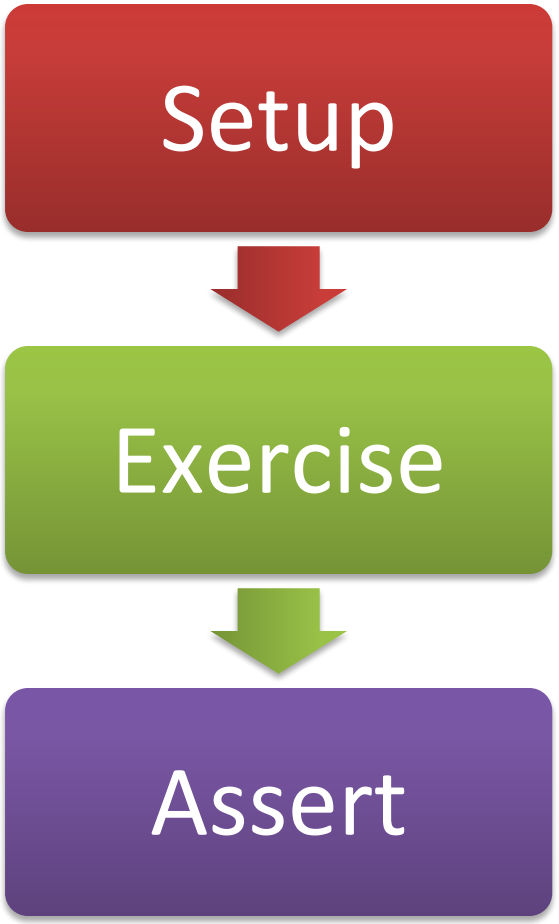
- Funcionales
- De Programador
- Automatizados, programados
- Preferentemente de Caja Negra
- Preferentemente Determinísticos
- Preferentemente Aislados
- Preferentemente “Sociales”

Características

- Deben correr rápido (< 10 ms x test)
- Deben tener las mismas características a nivel diseño, que cualquier otra pieza de software
  - Aunque a veces es conveniente repetir el “setup”, más aún si se puede reiniciar el test y se está trabajando con objetos mutables

Estructura de los Tests

Estructura de los tests



Creación de los objetos que se utilizarán como "datos de prueba" durante el test

Ejercita la funcionalidad específica que se está testeando. Determina QUÉ se está testeando.

Verifica que los resultados sean los esperados. Post-condición del test

10 Pines

Ejemplo

Datos de Prueba
vs.
Casos de prueba

Dato de Prueba != Caso de Prueba

- **Dato de Prueba**: Ejemplos concretos que “definen” un caso de prueba
- **Caso de Prueba**: Generalización que incluye los datos de prueba

10 Pines

Ejemplo

Lo que inicialmente nos parece un “Caso de Prueba” puede convertirse en “Dato de Prueba” al generalizar la solución

… por eso conviene esperar para nombrar los tests

“Inducción Incorrecta”

- **Inducción**: Para pasa para 1, pasa para N
  → Pasar para N+1

- **Inducción Incorrecta**: Pasa para 1, pasa para 2
  → pasa para N

Ejemplo

Necesaria porque es imposible testear con todos los datos de prueba

- Dijkstra: “Un test solo verifica que lo que se testea funciona o no”
- No es completo ni formal

Cómo nombrar los Tests

Ejemplo

Conclusión

- Nombrar los test en base al **caso de prueba** y no del dato de prueba
- Sintetizar en el nombre el “setup-exercise-assert”
- Los nombres de los tests son largos
- Usar *Given…When…Then…* cuando es difícil nombrar un test

- **RECORDAR**: Un test por caso funcional

El Test tiene que estar en “control de todo”

Ejemplo

Agrupar Aserciones

Ejemplo

Test de Excepciones

Ejemplo

Preguntas

Diseño ¡a la gorra!

¡Bienvenidos!
Durante esta serie de Webinars exploraremos *qué* significa **Diseñar Software con Objetos** y *cómo* lo podemos hacer cada vez mejor.
Trataremos muchos temas que irán desde cuestiones filosóficas como qué significa Diseñar en nuestra profesión y dónde está expresado ese Diseño, pasando por consejos y heurísticas para diseñar "mejor" y terminado con ejemplos concretos de cómo aplicar esas heurísticas en la *vida real.*
Los webinars son "*language agnostic*", o sea que no dependen de un lenguaje de programación en particular, aunque los ejemplos que usaremos estarán hechos principalmente en **Java**, **JavaScript**, **Ruby**, **Python** y mi querido **Smalltalk** cuando amerite 😉 .

Te esperamos todos los Martes a las 19 Hrs GMT-3 a partir del Martes 11 de Agosto de 2020.
Para poder participar tenes que registrate acá.

Todo el código y presentaciones estarán disponibles para que lo puedan usar y consultar en cualquier momento acá.

¡Trae ganas de aprender y pasarla bien!

¿Por qué a la Gorra?
Al igual que cuando Diseñamos Software está bueno usar una **Metáfora** para entender qué estamos modelando, en este caso usamos una metáfora para explicar cómo *financiaremos*

Donaciones

mercado pago

$100 - Casi una 🍷 Pagar

$250 - Una buena 🍺 Pagar

$500 - Menos que 🍔 +🍟 Pagar

PayPal Donate

¿Querés donar un monto variado o en otra plataforma?

Dictado por:
Hernán Wilkinson

https://alagorra.10pines.com

Muchas gracias

10 Pines

Creative Software Development

10pines.com

info@10pines.com

+54 (011) 6091-3125 / 4893-2057

Av. Leandro N. Alem 896 6° - Bs. As. - Argentina

@10pines