¿Porque hacemos “Testing”?

BY: ALFREDO ALVAREZ

Base para nuestra conversación

- Cual es el trabajo de un “tester”?
  - En el pasado-> Mantener la calidad y encontrar “Bugs”.
  - En estos días-> Mantener el equipo al tanto de todas las métricas que hacen falta para identificar el estado de un proyecto
- Porque el cambio?
  - La calidad que es la métrica mas importante contiene muchos mas elementos hoy en día que hace 10 años.

¿Que es un software “tester” ?

- Es aquella persona que pertenece a un grupo de desarollo encargado de una de las siguientes áreas:
  - Medir Calidad (Crear métricas y ejecutarlas)
  - Crear automatización de pruebas
  - Validar diseños sean flexibles (Keyword:Testability)
  - Encargarse del proceso automatizado de lanzamiento (Keyword:Continous Development)

¿Tipos de prueba para medir la calidad en el software?

- Black box "testing" – Se valida exactamente lo que el usuario ve de la forma que el usuario lo usaría.
  - Scenarios.
  - User acceptance tests.
- White box testing – Se valida el sistema empezando desde el componente mas oscuro y subiendo nivel por nivel hasta llegar a la interface.
  - Pruebas utilizando Apis
  - "Code Reviews"
  - Herramientas de Analisis Dinamico

Cualidades que se prueban.

- Regression Testing – Lo que funcionaba en la versión anterior todavía sirve?
- "Functional Testing" – Verificamos que la funcionalidad que esperamos este y que sea lo que espera el cliente.
  - ¿El producto puede sumar dos números con decimales?
  - Según el usuario nunca debe demostrar mas de dos cifras significativas.
- "Non Functional Testing"(ility) – Esta es la parte difícil.
  - Usability
    - ¿El producto se comporta de manera similar a otros en su categoría?
  - Testability
    - ¿Puedo probar todos los casos que le importan al cliente?
  - Scalability
    - ¿Cuántos usarios puedo usar por servidor?
    - ¿Tenemos un plan de capacidad?

Cualidades que se prueban.

- Availability
  - ¿Qué por ciento del tiempo deben estar nuestros servidores corriendo sin errores?
  - ¿Cuál es el minimo de maquinas que necesitamos para correr el sistema?
  - ¿Podemos instalar mas de una versión en los servidores al mismo tiempo?
- Security
  - ¿Esta definido nuestro sistema de authenticacion?
  - ¿Tenemos un checklist de seguridad que se esta aplicando?
- Etc - > Lo importante aquí es saber de antemano cuales nos importan en el producto actual.

Partes de una prueba

- Setup -> Como llegamos al punto que estamos listos para hacer una prueba
- Execution -> Que paso tenemos que ejercer para verificar nuestra prueba
- Assertion-> Cual es la contestación a nuestra prueba que determina el resultado
- Cleanup-> Como llevamos el sistema de vuelta al estado inicial.

Ejemplo de una prueba

1. Verificar que el programa de calculadora de Windows 8.1 pueda sumar decimales positivos.(Titulo de la prueba)
    1. Pre-requisito: Maquina tiene que tener una copia de Windows 8.1 con el programa calculadora recién abierto(Setup)
    2. Presionar 9.1 en el teclado
    3. Presionar la tecla de suma
    4. Presionar 0.4 en el teclado
    5. Oprimir = (Las naranjas son la ejecución)
    6. Resultado debe ser 9.5(Assertion)
    7. Cerrar programa de calculadora (Cleanup)

¿Qué pasa si no pasamos una prueba?

- Les presento a mi amigo el “bug”
- Un bug es simplemente un reporte de una prueba fallida
- Las partes mas importantes son las siguientes:
  - Titulo
  - Impacto
  - Pasos de Regresión (Como lo hicimos ocurrir).
  - Descripción
  - Status
  - Etc..(Muchos mas campos depende de la compañía)

Ejemplo de un bug

1. Titulo: Calculadora se cierra si oprimo ** dos veces seguidas.
2. Impacto: Alto(Es un error fácil de cometer y puedes perder los cálculos anteriores).
3. Descripcion:
   La calculadora termina abruptamente cuando se ejecutan los pasos.
4. Pasos de Regresión:
   1. Con la calculadora abierta
   2. Presione * dos veces en el teclado
   3. Vera un mensaje que contiene unhandled exception
5. Status: Abierto

Ya vimos una prueba cuales mas necesitamos?

- ¿ Podemos dividir?
- ¿ Podemos restar?
- ¿ Podemos multiplicar?
- ¿ Como se ve en high contrast mode?
- ¿ Cuantas calculadoras podemos abrir?
- ¿Funciona la nueva calculadora en todas las versiones de Windows?
- ¿Funciona con touch screen?
- Y continuamos hasta el infinito….

¿Cuando paramos de escribir y ejecutar pruebas?

- Realmente uno nunca lo puede probarlo todo.
- Lo importante es crear métricas que te dejen parar de escribir y de ejecutar.
- Estas métricas pueden salir como el resultado de diferentes documentos que describen un software.
  - En Waterfall
    - BRD- Business requirement document
    - FSD- Functional Specification document
  - En Agile
    - Stories
    - Scenarios

Ejemplos de métricas para el programa de la calculadora

- 90% de las pruebas usando multiplicación / división / resta y suma pasan
- La aplicación se puede utilizar perdiendo menos de 10% de la velocidad cuando se usa en un scenario de touch.
- En las ultimas dos semanas no hemos encontrado ningún bug.

Resumen

- En esta presentación nos enfocamos en la parte de calidad del aspecto de software testing.
- Vimos la importancia de las métricas
- Que las pruebas son las que componen las métricas
- Que un bug es un problema encontrado en una prueba
- Ejemplos de pruebas y métricas.

Referencias

- Software Testing (2nd Edition)
- Agile Testing: A Practical Guide for Testers and Agile Teams
- Lessons Learned in Software Testing: A Context-Driven Approach
- Whatever you do don't buy any James D. McCaffrey book(they are outdated).

- Enviame un email a:

instructorcodebytheneedle@gmail.com

Code By The Needle GeoRiot