

DEPARTAMENTO DE INFORMÁTICA

Ciclo Formativo: DESARROLLO DE APLICACIONES WEB Curso: 2º

Módulo: ENTORNOS DE DESARROLLO

Curso escolar: 2019-20

Unidad: 4-Documentación y Optimización de software

Evaluación: 2ª

Fecha aprobación: 22/01/2020

Fecha realización ejercicio: 22/01/2020

Práctica elaborada y aprobada por: Juan Antonio Gascón Sorribas

ÍNDICE

1. Introducción
2. Optimización
 1. Hediondez del código
 2. Análisis de código
 3. Análisis estático de código

Es un tipo de análisis que se realiza sin ejecutar el programa. Así, con esto, podemos detectar los siguientes errores:

 - Los problemas no funcionales del aplicativo.
 - La evidencia y prevención de problemas potenciales en etapas tempranas del ciclo de vida del software.
 - Todo ello sin pretender ocupar el lugar de una herramienta de pruebas de carga o el de una específica de seguridad.

Así, en resumen, con el análisis estático de código podemos anticipar posibles problemas antes de que se hagan realidad.

El término se aplica generalmente a los análisis realizados por una herramienta automática, el análisis realizado por un humano es llamado revisión de código. En la mayoría de los casos, el análisis se realiza en alguna versión del código fuente y en otros casos se realiza en el código objeto.

Algunas herramientas para este análisis son PMD (Program Mistake Detector), Checkstyle o Stylecop.
4. Linters
5. Inspección continua o análisis continuo. Continuous Inspection o continuous analysis
6. Refactorización
3. Documentación
 1. Insignias(Badges)
 2. Tipos de documentación
 1. Documentación de código
 2. Documentación técnica
 3. Documentación de usuario
 3. Formatos de Documentación
 1. HTML (Javadoc)
 2. Markdown (Gitbook)
 3. reStructuredText (Readthedocs)
 4. asciidoc
4. Control de versiones



1. Sistemas más conocidos
 1. CVS
 2. Subversion
 3. Mercurial
 4. Git
2. Git
 1. Conceptos
 2. Características
 3. Áreas
 4. Sitios que soportan GIT
 1. GitHub
 2. Bitbucket
 3. GitLab