Síntesis conceptual

Grado: Desarrollo de aplicaciones multiplataforma
Asignatura: Entornos de desarrollo
Unidad: 4. Optimización y documentación

Resumen

La programación es mucho más que la creación de contenido, un programador debe buscar mejorar siempre su trabajo, esto lo puede realizar mediante el empleo de las herramientas y prácticas apropiadas para ello, como las que se dan en este tema.

La refactorización es una de las técnicas más empleadas para la mejora y limpieza de código, pero este no es un proceso único, sino que existen una multitud de procesos con los que llevar a cabo esta labor que se engloban bajo la , algunos de ellos son:

- Extract method o reducción lógica.
- Métodos inline o código embebido.
- Variables autoexplicativas.
- Mal uso de variables temporales.
- Cambiar algoritmos.
- Mover métodos entre clases.
- Mover variables miembro entre clases.
- Autoencapsular campos (getters y setters).
- Extraer clases.
- Reemplazar valores con objetos.
- Empleo de constantes.
- Encapsular arrays.
- Reemplazar tipos de objeto con subclases.

Otro de los procesos que podemos emplear para para mejorar nuestro código es el uso de patrones de diseño, los cuales nos permitirán evitar de una forma muye sencilla y eficiente errores comunes. Podemos encontrar diversos tipos de estos.

- Patrones de creacionales: Ayudan en la creación de nuevos objetos para los POO.
- Patrones estructurales: Permite evitar problemas en las relaciones entre objetos.
- Patrones de comportamiento: Ayuda a encontrar los documentos entre objetos y a modificarlos.

La refactorización produce cambios en el código, lo cual provoca que se generen nuevas versiones del código. Este constante cambio de documento puede llevar a que se produzcan errores durante los cambios, especialmente problemáticas son las incompatibilidades que se pueden producir al añadir nuevos fragmentos al código ya escrito.

Con el fin de poder llevar a cabo la implementación de nuevas versiones sin miedo a estropear el resto del código los SCV son imprescindibles. Los SCV, sistemas de control de versiones, nos permiten guardar versiones antiguas de nuestro programa, de modo que, ante cualquier problema surgido con una nueva versión, podamos retroceder fácilmente y sin problemas a una versión anterior y descubrir el problema antes de volver a implementarlo.

Este tipo de sistemas también nos permite distintos modos de trabajo cuando se requiere el trabajo colaborativo.

Por último, una parte importante de nuestro trabajo es la documentación, la cual debe ser no solo completa sino de calidad, para los cual es necesario seguir una serie de recomendaciones:

- Realizar esquemas con los que estructurar nuestras ideas.
- Clasificar y diferenciar la información relevante de la innecesaria.
- Elaborar síntesis, índices y esquemas de nuestro trabajo.
- Uso de elementos estándar.
- Anticipar las dudas y problemas de los lectores.
- Expresarse con claridad y con un lenguaje apropiado.
- Empleo de las herramientas adecuadas para la comunicación, ya sea un procesador de texto o uno de imágenes.
- Empleo de los documentos apropiados para cada una de las situaciones.
- Adaptar el lenguaje al nivel del lector.

Conceptos fundamentales

- **Refactorización**: proceso de reestructuración interna de un código sin cambiar el programa.

- **Setters**: de *set*, establecer, permite determinar el valor inicial de un atributo.

- **Getters**: de *get*, obtener, permite recuperar el valor de un atributo.

- **Patrones**: son plantillas de código con soluciones a problemas generales.

- ***Workflow***: modo de trabajo que lleva al desarrollo de este en un orden y de una manera específica. Una manera de trabajar específica de una empresa, grupo o persona.