Diseño de gráficos para dispositivos móviles

La visualización de la información es un área de la computación que trata del desarrollo de representaciones visuales interactivas que facilite a las personas el descubrimiento de patrones y el análisis de la información, para ello se pueden emplear elementos visuales como cuadros, gráficos y mapas, las herramientas de visualización de datos proporcionan una manera accesible de ver y comprender tendencias, valores atípicos y patrones en los datos.

La visualización de datos es otra forma de arte visual que capta nuestro interés y mantiene nuestros ojos en el mensaje. Cuando vemos un gráfico, vemos rápidamente las tendencias y los valores atípicos. Si podemos ver algo, lo interiorizamos rápidamente. Es contar historias con un propósito. Si alguna vez vemos una gigantesca hoja de cálculo de datos y no nos es posible ver una tendencia, sabemos cuán eficaz puede ser una visualización.

1.1 Objetivo general

Emplear la herramienta Flutter para diseñar gráficos que permitan comunicar información, mostrar y comparar variables.

1.2 Descripción general

Deben diseñar una visualización en la herramienta Flutter para mostrar y comparar variables acerca de la población con discapacidad en Costa Rica. Para la población con discapacidad las variables son el tipo de discapacidad, sexo, grupos de edad y provincia.

Debe realizar un gráfico de burbujas, las burbujas son un indicador visual de magnitud. Adicionalmente, debe tener la opción de que el usuario pueda seleccionar otros dos tipos de indicadores visuales de magnitud, por ejemplo, rectángulo (en vez de que sea circular que sea rectangular), barra, glifo, entre otros.

En su diseño debe analizar opciones para los elementos gráficos tales como: colores, fuente y tamaños.

1.3 Actividades

a) Estudiar Flutter y determinar posibles bibliotecas a utilizar.

a) Definir su objetivo (qué es lo que haría que su visualización sea funcional y eficiente para que un usuario entienda y aproveche la información que va a mostrar).

b) Llevar a cabo actividad de diseño visual, lo cual implica hacer bosquejos, analizar opciones y definir:
    - Layout: distribución de elementos en la pantalla.
    - Características visuales de los indicadores de magnitud.
    - Fuente.
    - Tamaños.
    - Opciones de interactividad y animación.
    - Propuesta de forma en que el usuario puede escoger la configuración.
c) Programar, probar y afinar la visualización.
d) Revisión:
    - Describir qué hace a su diseño diferente y por qué considera que es un buen diseño.
    - Justificar cada decisión de diseño visual.

1.4 Control de versiones

Una de las tareas diarias de todo desarrollador de software en un proyecto es mantener un reporte de las nuevas características, mejoras, arreglos y otras modificaciones. Para facilitar esta tarea, cada grupo utilizará *git* (https://git-scm.com), que es una herramienta para el control de versiones muy popular debido a su rendimiento, extensión y documentación.

A continuación, se explicará el flujo de uso que se adoptará para cada proyecto de software, basándose en la metodología GitFlow (http://nvie.com/posts/a-successful-git-branching-model).

1.4.1 Commits

El desarrollador o responsable de realizar un commit debe procurar que esta acción responda a una de las siguientes razones:

- Se ha terminado toda o una parte importante de la funcionalidad.
- Se está respaldando el trabajo realizado antes de cambiar a otra tarea.

El desarrollador debe hacer una descripción puntual sobre la parte de la funcionalidad que está respaldando en el commit. Sobre los mensajes de commit:

- Procurar que el mensaje sea conciso sobre el cambio implementado, preferiblemente una sola frase.

- Se procurará que sean en idioma inglés.

Se sugiere escribir una de las siguientes etiquetas al principio de la descripción del mensaje, de acuerdo con el tipo de cambio que realiza un commit:

- [new]: se ha creado un método o recurso en el programa que no existía antes del commit.
- [improved]: se mejoró la forma en que se hacía un método o como se mostraba algo. No es un problema como tal.
- [fixed]: se corrigió un problema, algo que estaba mal.
- [updated]: se reemplazó un recurso o código por otro realizado por alguien más (ejemplo: una biblioteca de funciones).
- [init]: commit especial que indica el inicio del repositorio.
- [incomplete]: commit especial que indica que se ha respaldado una versión inestable o incompleta de la funcionalidad. Se recomienda usarlo como una forma de indicar un respaldo temporal del trabajo hacia una versión estable de la funcionalidad que se está programando, pero que se tiene que dejar por un momento. Ejemplos: parar lo que se está haciendo para atender un hotfix, seguir el trabajo en otro equipo de cómputo. Se recomienda utilizar luego la opción ammend para sobrescribir este commit por uno de los normales, de acuerdo con el tipo de cambio del commit.

1.4.2 GitFlow

GitFlow es una metodología de trabajo para organizar y flexibilizar el trabajo paralelo en Git por medio de los branches. Su objetivo es aislar versiones estables del programa en producción de las versiones que se están en desarrollo, tomando en cuenta las etapas de pruebas y correcciones de emergencia (hotfixes).

Branches de desarrollo y producción

En GitFlow, existe un branch develop sobre el cual estarán las últimas características estables, listas para ser probadas. El branch master contiene la última versión publicada del programa, que el usuario final está utilizando.

Branches derivados de develop

Los programadores trabajarán las nuevas características sobre branches derivados de develop, llamados features. Sólo se podrá trabajar directamente el branch develop si se realiza una

modificación de poco impacto para la funcionalidad del proyecto y que no se considere como una nueva característica mayor.

Branches features para agregar nuevas características y mejoras

Se creará un branch a partir de develop cada vez que se requiera implementar:

- Una nueva funcionalidad mayor del programa
- Una mejora de una funcionalidad existente
- Una corrección no urgente y de medio a gran impacto

El formato para el nombre del branch creado será **feature/nombre-de-la-funcionalidad**, independientemente de si se trata de una funcionalidad o un ajuste/corrección, como se muestra en el siguiente ejemplo:

- Branch: feature/debugSpeak
    - [New] Speech synthesis debugging
- Branch: feature/issues
    - [Updated] Edit options (onActivated)
    - [Updated] Upper tool bar
    - [Updated] Scrollbar position (symbols)
    - [Updated] Container position
- Branch: feature/menubar
    - [Fixed] Focus keep on popup when is closed
    - [Fixed] Shortcut alignment
    - [Fixed] Up key on deutsch button go to french button

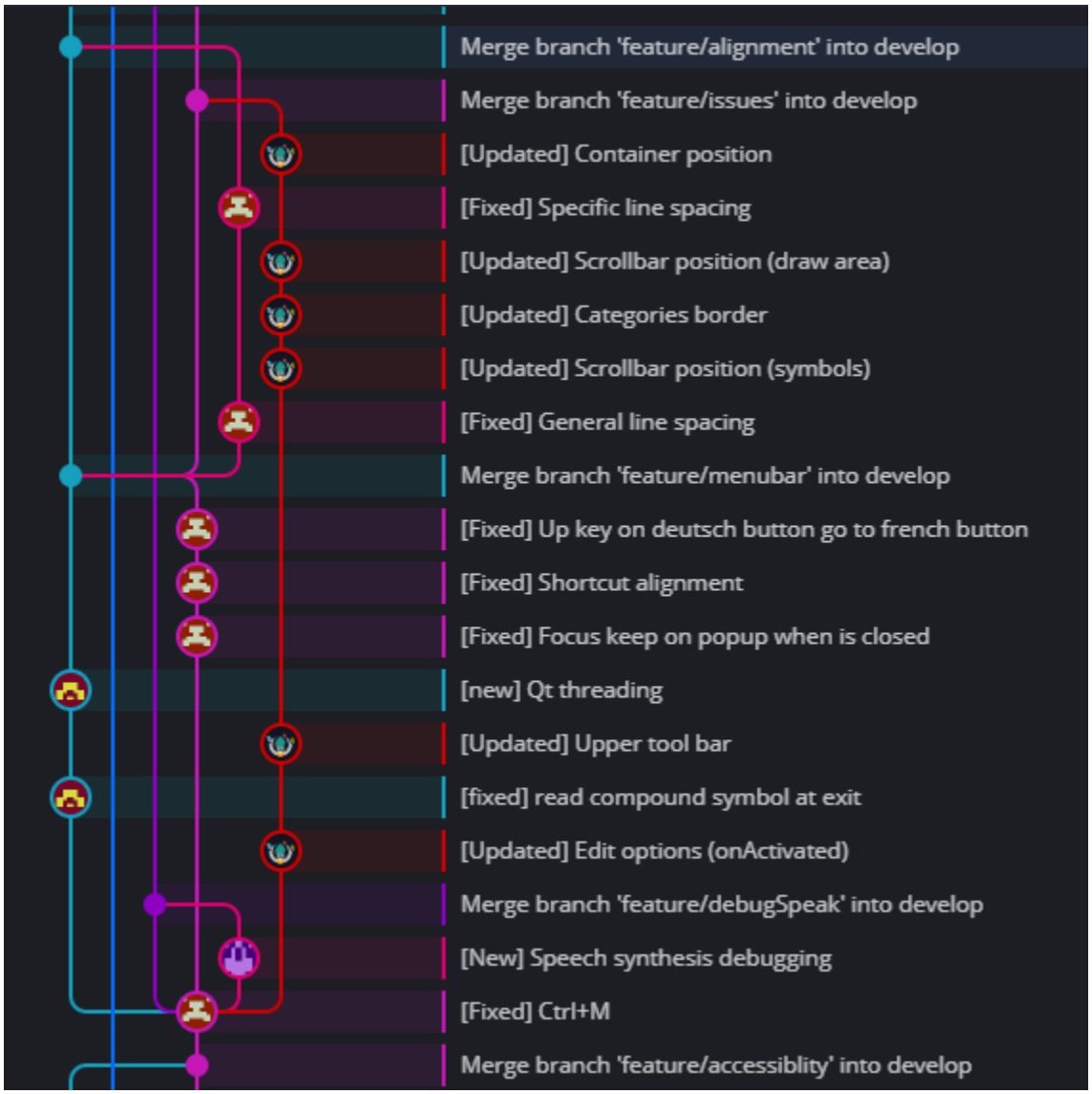

Figura 1: Ejemplo de commits y branch

(Fuente: Elaboración propia)

1.5 Aspectos administrativos

- Fecha límite de envío del repositorio (por correo) **jueves 2019.03.19, 23:59**
- El proyecto corto es en parejas.
- El asunto de sus mensajes de correo relacionados con este curso *debe tener siempre el prefijo*, "**[IC-8056]**". Ese prefijo debe ser seguido por "PC **1**" – Nombre de los estudiantes.