# RESUMEN DDI EXAMEN 1

Interfaz de usuario: Es el medio por el cual un usuario puede comunicarse con una máquina, equipo, ordenador, dispositivo, etc… Y comprende todos los puntos de contacto entre el usuario y el equipo. Las más básicas incluyen menús, botones, etc…

Los principales factores a la hora de desarrollar una interfaz son:

- Simplicidad: Interfaz sencilla, clara y concisa. Lo mejor es hacer una jerarquía.
- Experiencia: Se debe ofrecer una experiencia sensorial al usuario
- Detalle: Todo usuario aprecia los pequeños detalles

Todo usuario requiere de un tiempo de adaptación a una interfaz. Una buena interfaz se vuelve invisible al interactuar con ella. Algunas de las cosas que pueden hacernos sentir perdidos son:

- Mensajes poco claros
- Encontrar la información es complicado
- No existe jerarquía alguna

El diseño de interfaces o ingeniería de la interfaz es el resultado de definir la forma, función, usabilidad, ergonomía, imagen, etc… que afectan a la apariencia externa de las interfaces de usuario en sistemas de todo tipo.

Es una actividad multidisciplinar, involucra ramas como diseño gráfico, diseño web, diseño industrial, etc…

----------**PROCESO DE DISEÑO DE UNA INTERFAZ**----------

- Ingeniería de requisitos: Elaboración de una lista de elementos funcionales requeridos por el sistema para cumplir con las necesidades del proyecto y de los usuarios
- Análisis de usuarios y tareas que van a realizar: Análisis de los usuarios potenciales, estudio de la forma en la que realizan las tareas y realización de entrevistas. Preguntas frecuentes como ¿Qué querría el usuario que haga el sistema? Etc…
- Inspección de la usabilidad: Un evaluador externo debe inspeccionar la interfaz. Dicho evaluador no tiene que tener ni idea de como es la interfaz y debe de tener un conocimiento básico de usabilidad web

5 objetivos clave:

- SIMPLEZA
- CLARIDAD
- COHERENCIA
- FAMILIARIDAD
- RAPIDEZ

## DIFERENCIAS CLAVE ENTRE USUARIOS ADULTOS E INFANTILES

Los niños aprecian positivamente los efectos de sonido y animaciones, además de interfaces ricas en colores, así como una navegación sencilla y con poco texto. Además, los niños suelen recorrer la interfaz con el ratón en busca de elementos de sonido o animaciones, suelen hacer clic en los banners y anuncios ya que no lo perciben como información diferente y la atención de los niños es más alta cuando en la interfaz se encuentra algún personaje de animación conocido o algo del estilo.

Los niños prefieren tener una pequeña guía antes de realizar alguna actividad o juego, mientras que los adultos prefieren saltar directamente a la actividad, sin leer las instrucciones. Los adultos suelen usar aplicaciones orientadas a temas laborables o para tareas que tienen objetivos, mientras que los niños utilizan aplicaciones para entretenerse

## USABILIDAD

La usabilidad se refiere a la facilidad a la hora de utilizar una aplicación interactiva. No todas las aplicaciones se pueden diseñar utilizando los mismos criterios de usabilidad. Se compone tanto de atributos objetivos (tiempo empleado para conseguir un objetivo, numero de errores, etc.) como de atributos subjetivos (satisfacción del usuario)

## PRINCIPIOS GENERALES DE USABILIDAD

- Visibilidad del estado del sistema: El sistema mantiene informado al usuario sobre el estado del mismo, indicadores que sirven de feedback y de fácil lectura
- Consistencia: Sistema homologado, lenguaje, colores, etc. Consistentes
- Control de usuario: El sistema debe adaptarse al usuario, apoyándolo en vez de entorpeciéndolo
- Prevención de errores: Todos cometemos errores y debemos hacer que los impactos sobre el sistema sean lo menos consecuentes posible. Hacer casos de prueba poco comunes y probar nuestros algoritmos a fondo harán un buen sistema
- Estructura visible: Índices o mapas que representen la estructura del sistema, permitiendo la fácil navegación entre la misma
- Interfaz explorable: La interfaz debe estar diseñada para que el usuario sepa que rutas existen y pueda llegar a cualquier punto del sistema
- Ley de Fitts: Las opciones más importantes deben tener mayor tamaño o ser más visibles. La localización es importante también, los 4 lados de una ventana suelen ser los puntos más accesibles
- Modalidad: Los modos sirven para contextualizar temporalmente las acciones del usuario. Cuadros de dialogo que paren todo el sistema hasta que el usuario responda

- Metáforas (iconos): Elementos que se asemejan a los del mundo real
- Uso del color: Los colores se pueden usar para varios propósitos, como captar la atención del usuario, distinguir elementos, organizar información, etc… Debemos hacer un buen uso del color para que la interfaz cumpla con los objetivos mencionados antes
- Mensajes de error: Prevención, para reducir el numero de mensajes de error. En caso de que salgan igualmente, el mensaje debe describir el problema BREVEMENTE. Basta con indicar al usuario cuales son las opciones para solucionar el error. Evitar la palabra 'error', no utilizar mayúsculas ni exclamaciones, mensajes auditivos limitados y todas las ventanas de error deben tener una opción CLARA para poder cerrar dicha ventana
- Tiempos de respuesta: La respuesta del sistema ante las acciones del usuario debe ser rápida e inmediata. Se utilizan señales visuales o auditivas para mostrar al usuario que la acción se ha detectado y se está procesando

- Todas las opciones de menú, iconos, etc. deben ofrecer una **respuesta visual inmediata.**
- El usuario ha de poder **cancelar cualquier proceso.**
- Es aconsejable que aparezca un **indicador gráfico de espera (por ejemplo,** un reloj) para cualquier acción que pueda durar más de ½ segundo. Este indicador debe estar animado para que no parezca que el sistema está bloqueado.
- Debe aparecer un mensaje que indique la **duración para cualquier proceso** que dure más de 2 segundos.
- El **indicador de progreso puede mostrar el estado del proceso mediante** una barra animada u otro recurso similar. Puede acompañarse de un indicador numérico de porcentaje.

25

## CONSISTENCIA

1) Acciones: El usuario debe poder repetir la misma secuencia para acciones similares

2) Terminología: Conceptos utilizados deben ser los mismos en todo el sistema

3) Elementos gráficos: Retícula, gama de colores, etc… deben mantenerse constantes en todo el sistema.

# LA PLATAFORMA .NET de MICROSOFT

Plataforma de desarrollo de código abierto, multiplataforma y gratuita para crear diferentes tipos de aplicaciones

NO es un sistema operativo. NO es un lenguaje de programación. NO es un entorno de desarrollo. NO es un servidor de aplicaciones.

Competencia a Java de Oracle por parte de Microsoft en el sector de desarrollos web

Permite desarrollar:

- Aplicaciones de consola
- Windows Forms (aplicaciones con interfaz gráfica)
- Aplicaciones WPF (Igual que la anterior, pero con la interfaz gráfica enriquecida)
- Aplicaciones ASP.NET (aplicaciones web)
- Aplicaciones Silverlight (aplicaciones web enriquecidas)
- Servicios Windows y servicios web

Algunos de los beneficios que proporciona son:

- Disminución del tiempo de desarrollo
- Reutilización de funcionalidades ya diseñadas
- Simplifica el mantenimiento de las aplicaciones
- Reduce los costos debido a todo lo anterior

.NET dispone de unas implementaciones que manejan el trabajo pesado:

- .NET Framework: implementación normal, compatible con servicios web, aplicaciones de escritorio, servicios, pero solo en Windows
- .NET Core: Implementación multiplataforma, lo mismo que la anterior, pero en Windows, Linux y MacOS
- Xamarin: Aplicaciones principales en dispositivos móviles. Tanto iOS como Android
- .NET Standard Library: API común a todas las implementaciones anteriores

## COMPONENTES DE .NET FRAMEWORK

- CLR (Common Language Runtime): Motor que controla la ejecución de las aplicaciones. Es una máquina virtual.
- Biblioteca de clases .NET: Proporciona una biblioteca de código válido para su reutilización en el desarrollo de aplicaciones
- Biblioteca ADO.NET: Proporciona acceso a Bases de Datos

## ARQUITECTURA .NET

- CLS (Common Language Specification): Define las características fundamentales del lenguaje. Define un conjunto de datos comunes. Esto permite usar varios lenguajes de programación
  Por ejemplo, en VB, un numero entero se declara con integer. En C# como int. Estas 2 sintaxis son sinónimas del tipo común System.Int32

## COMO FUNCIONA .NET

El proceso de ejecución se divide en varios pasos:

1. Diseñar y escribir el código fuente (Visual Basic, C#, C++, …)
2. Compilar el código fuente a código intermedio (generar un .exe)
3. Compilar el código intermedio a código nativo (lenguaje máquina)
4. Ejecutar el código nativo.