Introducción a la programación con C#

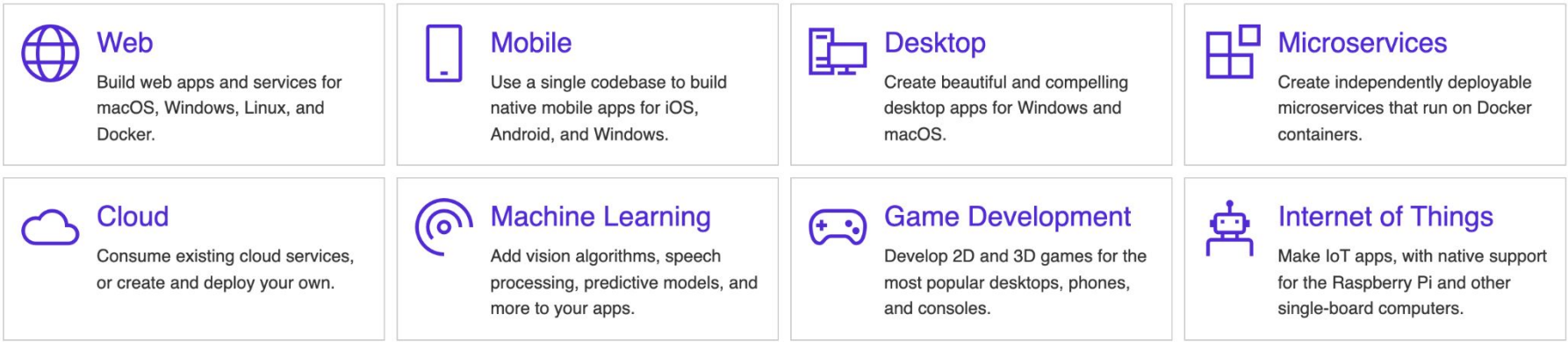
Introducción al lenguaje C# y Algoritmos

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
Centro de e-Learning

¿Qué vamos a ver hoy?

- Entender el ecosistema .NET
- Instrucciones básicas de programación
- Primeros conceptos de POO
- Como crear un proyecto desde Visual Studio
- Diferentes tipos de proyectos

.NET



Desde acá, podés acceder a la web oficial y bajarte la última versión del framework

¿Qué es la POO?

Es un paradigma de programación. En el que los objetos se utilizan como metáfora para representar las entidades reales de lo que queremos modelar.

Bases o conceptos:

- Abstracción
- Encapsulación
- Modularización
- Jerarquización

Abstracción

Es el proceso mental de extracción de las características esenciales de algo, ignorando los detalles superficiales.

Coche alquilado:

- La tarifa
- Limitación de kms
- ¿Consumo?

Coche particular:

- Consumo
- El color
- Valor de reventa
- Duración de los componentes

Coche para repartos:

- Consumo
- Carga soportada
- Garantía

Encapsulación

Es el proceso en el cual se ocultan los detalles del soporte de las características de una abstracción.

No se trata de ocultar las características en sí, sino de no mostrar cómo guardo o maneja internamente esas características.

Por ej, de la clase fecha nos interesa obtener el día, mes y año. Y no como esta guardado.

Modularización

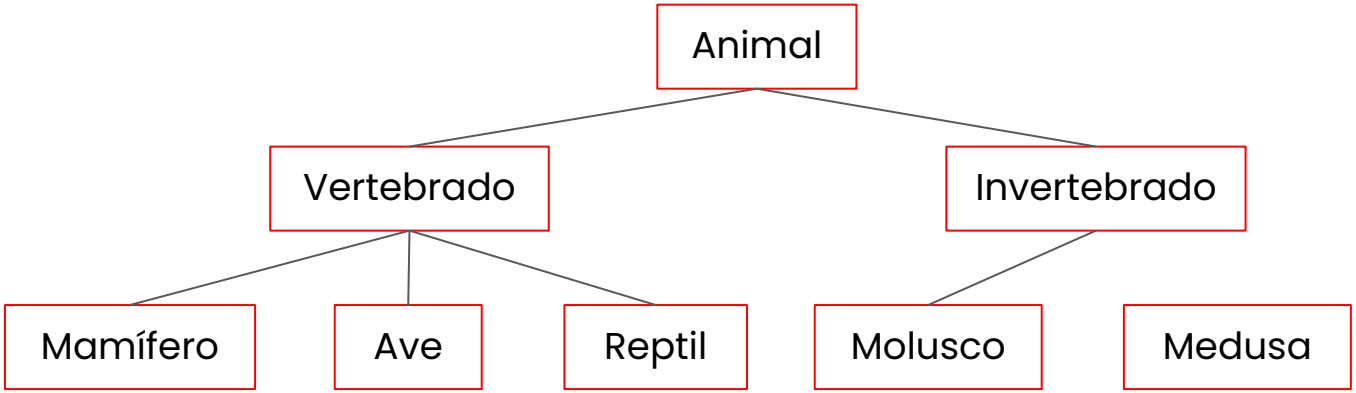
Es el proceso de descomposición de un sistema en un conjunto de módulos o piezas.
Que esten poco **acopladas** (independientes) y **cohesivas** (con significado propio).

Se entiende por sistema al conjunto de piezas que colaboran entre sí.

Por ej, el módulo de cierre centralizado con el módulo de la alarma del auto.

Jerarquización

Es el proceso de estructuración por el que se produce una organización de un conjunto de elementos en niveles de responsabilidad.



Elementos

- Clase: descripción de los datos y de las operaciones de un elemento modelado.
- Objeto: instancia de una clase, ejemplar concreto.
- Mensaje: invocación de una operación sobre un objeto.
- Método: definición de una operación de una clase.
- Atributo: los datos de una clase, y por tanto, presente en todos los objetos de esa clase.
- Estado: conjunto de los valores de los atributos que tiene un objeto.

Relación de elementos

Una **clase,** es la definición de los **atributos** y **métodos** que describen el comportamiento de un **objeto**.

Un **objeto**, es un ejemplar concreto de una **clase** que responde a los **mensajes** correspondientes a sus **métodos**, adecuándose al **estado** de sus **atributos**.

Palabras claves

Son identificadores reservados predefinidos que tienen significados especiales para el compilador.
Estos no se pueden utilizar como identificadores en nuestro programa.

Algunos ejemplos:

- abstract
- break
- continue

Documentación oficial

Modificadores de visibilidad

Todos los tipos y miembros de tipo (clase, método, atributo) tienen un nivel de accesibilidad. Controla si se pueden usar desde otro código en su ensamblado o en otros ensamblados.

- public: puede ser accedido desde cualquier código.
- private: puede ser accedido desde la misma clase.
- internal: puede ser accedido desde el mismo ‘assembly’.

Documentación oficial

Sentencias de flujo de ejecución

Son aquellas sentencias que nos permiten manipular nuestro código. Que, dependiendo determinado criterio, nuestro código salte a una u otra línea.

- If
- Switch
- For
- Foreach
- While
- Do While
- Try & Catch

Arquitectura **M**odel-**V**iew-**C**ontroller

Se utiliza para desacoplar la interfaz de usuario (vista), los datos (modelo) y la lógica de la aplicación (controlador).

Separa una aplicación en tres grupos de componentes principales: **modelos**, **vistas** y **controladores**.

Permite lograr la separación de intereses.

Arquitectura **M**odel-**V**iew-**C**ontroller

Las solicitudes del usuario se enrutan a un **controlador** que se encarga de trabajar con el **modelo** para realizar las acciones del usuario o recuperar los resultados de consultas.

El **controlador** elige la **vista** para mostrar al usuario y proporciona cualquier dato de **modelo** que sea necesario.

¿Qué es una API?

Application **P**rogramming **I**nterface

Es una interfaz que permite la comunicación entre 2 aplicaciones.

Podemos decir que es un sitio que en vez de responder algo visual, como HTML con CSS, nos responde información.

Los usuarios probablemente no van a ingresar a este tipo de sitios, pero sí una aplicación que necesite abastecerse de cierta información.

Por ejemplo, una aplicación que necesite un listado completo de todos los países del mundo podría consumir una API que le otorgue esa información.

¿Y una API REST?

REpresentational **S**tate **T**ransfer

Es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP.

Proporciona un fuerte aislamiento con quien la consume.

Los objetos REST son manipulados a través de una URI (Uniform Resource Identifier).

Esta URI (endpoint) hace de identificador único de cada recurso del sistema REST, por lo que no puede ser compartida por más de un recurso.

REpresentational **S**tate **T**ransfer

La estructura básica de una URI es la siguiente:

{protocolo}://{hostname}:{puerto}/{ruta del recurso}?{parámetros de filtrado (opcional)}

El nombre de la URI no debe contener palabras que impliquen acciones, por lo que deben evitarse los verbos en su construcción.

Además, las URI siguen una jerarquía lógica de capas que permite ordenar los recursos y englobar las distintas funcionalidades entre sí.

Métodos HTTP

Los métodos son usados para manipular los diferentes recursos que conforman una API.

Los principales métodos soportados por HTTP son:

- POST: crear un recurso nuevo.
- PUT: modificar un recurso existente.
- GET: consultar información de un recurso.
- DELETE: eliminar un recurso determinado.
- PATCH: modificar solamente un atributo de un recurso.

Métodos HTTP

Estos métodos, junto con la URI, nos proporciona una **interfaz uniforme** que nos permite la transferencia de datos aplicando operaciones concretas sobre un recurso determinado.

Aunque la mayoría de las operaciones que componen una API REST podrían llevarse a cabo mediante métodos GET y POST, el abuso de ellos para operaciones que nada tienen que ver con el propósito con el que se concibieron, puede provocar un mal uso.

¡¡A trabajar!!