



**Escuela Politécnica Superior de
Alicante**

Diseño de aplicaciones web

Introducción al
Diseño de aplicaciones Web

Índice

- Sitios web
- Aplicaciones de internet
- Frameworks de desarrollo de SW
- MVC
- Visual studio
- ASP.net
- ASP.net MVC
- ASP.net CORE

¿Que es la web?

- Wikipedia:
 - la World Wide Web (WWW) o red informática mundial, es un sistema que funciona a través de Internet, por el cual se pueden transmitir diversos tipos de datos a través del Protocolo de Transferencia de Hipertextos o HTTP
- En otras palabras:
 - La web es un servicio de la red de internet que consiste en la transferencia de recursos entre servidores y clientes: documentos (páginas HTML), imágenes, etc.

¿Que es la web?

- **Tecnologías en las que se basa la web:**
 - HTTP: Protocolo de transmisión de documentos web y ficheros asociados.
 - HTML: Formato de diseño de documentos web que permite un maquetado flexible y completo.
 - URI/URL: Sistema que permite que todo documento HTML o recurso en la web pueda ser localizado a partir de una dirección única
- **La idea: Hipertexto, texto en 3D.**
 - El contenido de una página web puede enlazarse con otras de forma que su lectura ya no es secuencial (como un texto normal) sino que puede saltarse de un documento a otro.

HTTP

- Hypertext Transfer Protocol
- Protocolo de transmisión de documentos web y ficheros asociados.
- Pertenece al conjunto de protocolos de internet
- Trabaja a nivel de aplicación
- Creado por Tim Berners-Lee en 1989.
- Tipo petición-respuesta en entorno cliente/servidor
- Sin estado: stateless protocol

HTML

- Hypertext Markup Language
- Formato de diseño de documentos web que permite un maquetado flexible y completo.
- Deriva de SGML
- Creado por Tim Berners-Lee en 1989
- Una página HTML es una propuesta de representación del documento. La visualización final dependerá del contexto: navegador (marca, versión), pantalla (resolución, proporciones), sistema operativo, etc.
- HTML5: versión actual que además de HTML integra varias tecnologías accesorias (CSS, Javascript, etc.)

URL

- Uniform Resource Locator → dirección web.
- Sistema que permite que todo documento HTML o recurso en la web pueda ser localizado/accedido a partir de una dirección única.
- Se basa en un sistema preexistente de nombres de dominios (DNS)
- Formato:

protocolo://servidor/ruta/recurso?consulta#fragmento

- Donde:
 - Protocolo: http, https, ftp, file, etc.
 - Servidor:
[usuario@]dominio1.dominio1.dominio1nivel[:puerto]
 - Ruta: directorio1/directorio2/...
 - Recurso: nombre recurso o fichero (extensión opcional)
 - Consulta: nombre1=valor1&nombre2=valor2&...

¿Que es un sitio web?

- Punto de vista del usuario
 - Un conjunto de páginas web agrupadas bajo un mismo tema o entidad y accesibles en un **mismo dominio**.
- Punto de vista del diseñador
 - Un conjunto de documentos HTML con la misma imagen y disposición de elementos en pantalla cuyo contenido tiene un escenario o un aspecto homogéneo (imagen corporativa).
- Punto de vista del analista/programador
 - **Un aplicativo que funciona sobre entorno cliente/servidor (internet) principalmente para consultar datos contra una BD. Opcionalmente puede tener un gestor de contenidos.**
- Punto de vista del administrador de sistemas
 - Un conjunto de ficheros publicados en un servidor web con distintos permisos y opciones.

¿Que es un sitio web?

- ¿Sinónimos?
 - Sitio web
 - Página web
 - Aplicación web
 - Intranet
 - Aplicación en Internet

Sitios web: primera generación

- El servicio web comenzó a funcionar en 1990.
 - Autor: Tim Berners-Lee
- Originalmente la web estuvo orientada a la comunicación y transmisión de documentación científica.
- Los sitios web eran texto, tablas e imágenes (pocas y de baja resolución)
- El contenido de los sitios web estaban determinados por las limitaciones de:
 - Hardware (poca resolución y colores)
 - Comunicaciones (baja velocidad, poco ancho de banda)
 - Software: pocos navegadores, limitados.

Sitios web: primera generación

World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#), [Policy](#), November's W3 news, [Frequently Asked Questions](#).

What's out there?

Pointers to the world's online information, subjects, W3 servers, etc.

Help

on the browser you are using

Software Products

A list of W3 project components and their current state. (e.g. [Line Mode](#), [X11 Viola](#), [NeXTStep](#), [Servers](#), [Tools](#), [Mail robot](#), [Library](#))

Technical

Details of protocols, formats, program internals etc

Bibliography

Paper documentation on W3 and references.

People

A list of some people involved in the project.

History

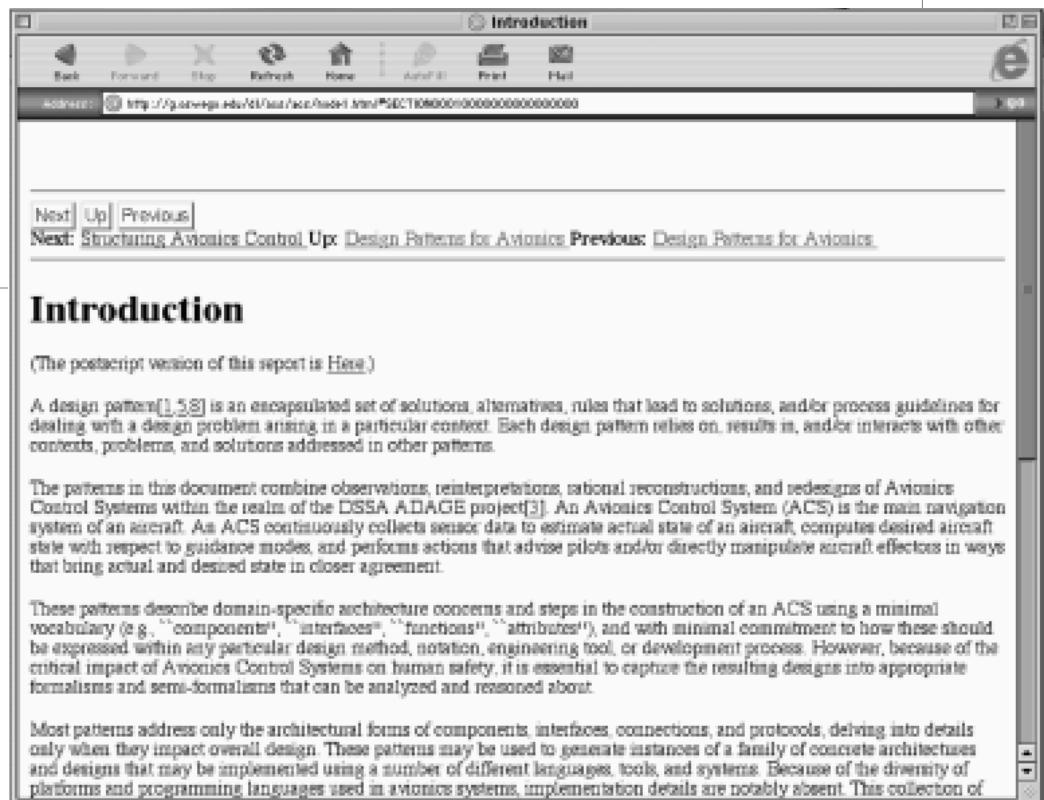
A summary of the history of the project.

How can I help ?

If you would like to support the web..

Getting code

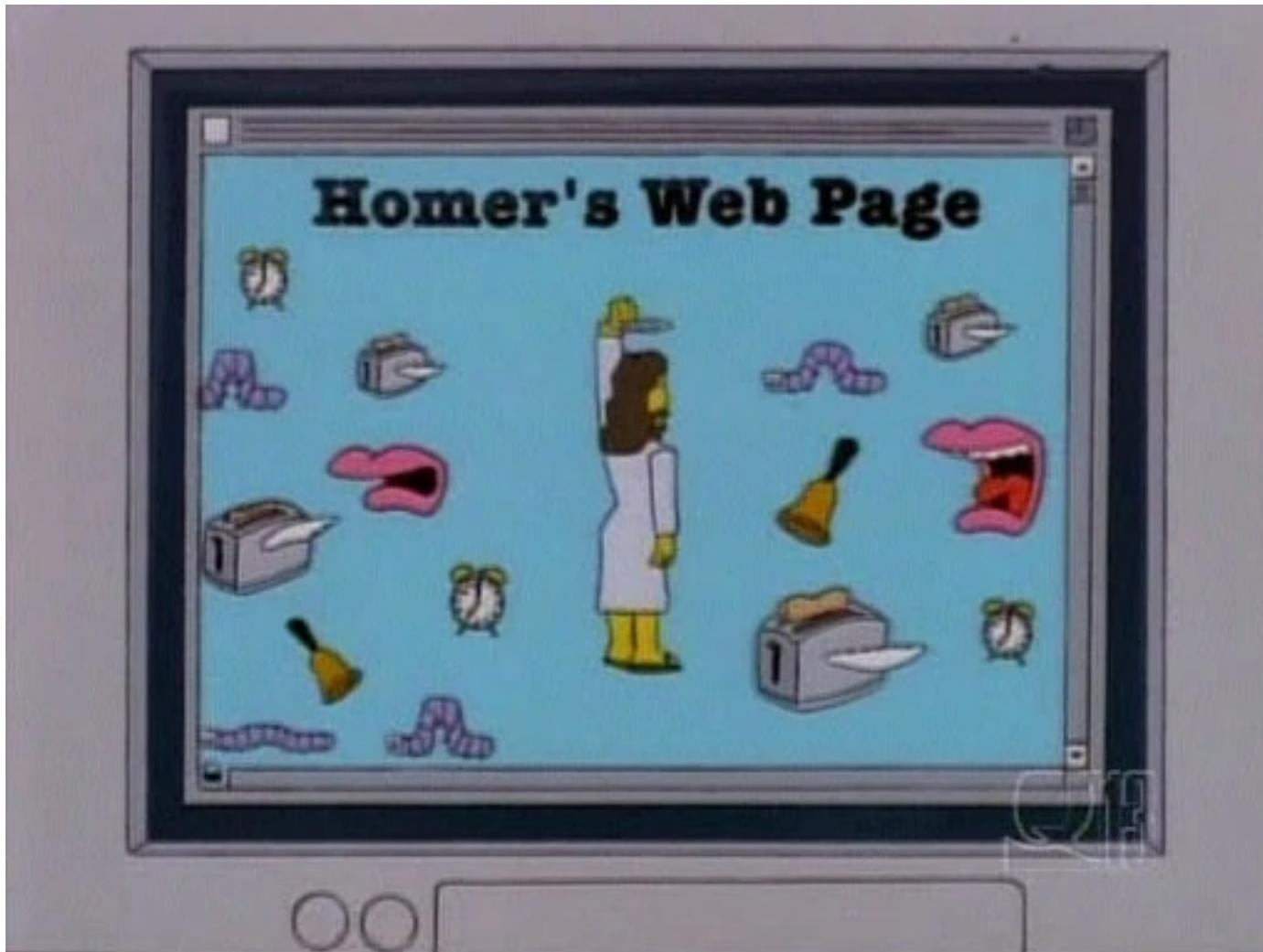
Getting the code by [anonymous FTP](#), etc.



Sitios web: segunda generación

- Mejora en el software: aparecen navegadores de distintas marcas que soportan más prestaciones.
- Las páginas web salen del entorno universitario y comienzan a llegar a ámbitos gubernamentales, personales y, en menor medida, empresariales.
- Mejora de las comunicaciones que provoca un mayor uso de imágenes y colores.
- Programación de servidor únicamente con scripts ejecutados en el servidor mediante la pasarela CGI.
 - Shell scripts, C, PERL, AWK

Sitios web: segunda generación



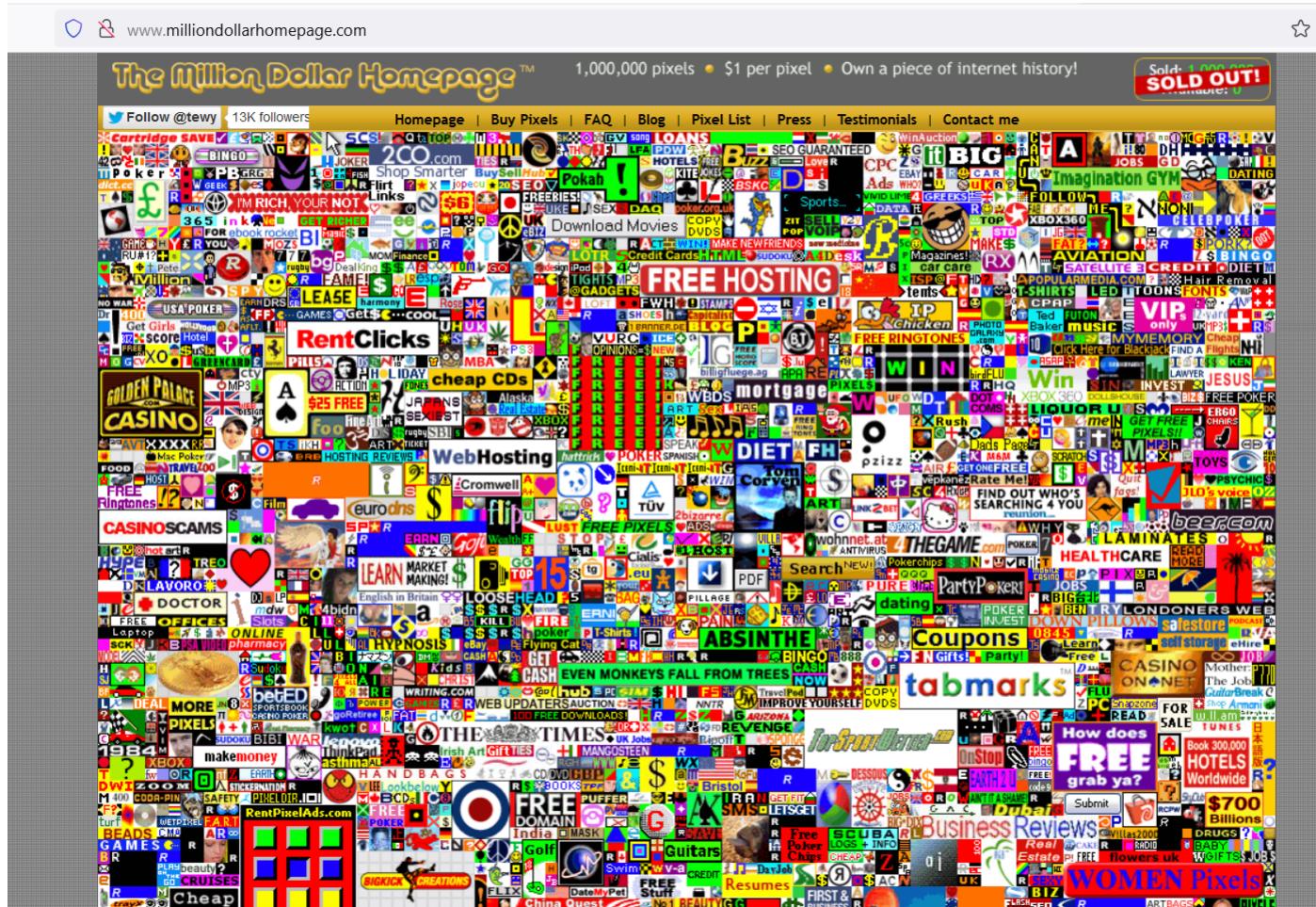
Sitios web: tercera generación

- Aproximadamente a partir del año 1996 con la aparición del Netscape 3.0.
- Primeras tecnologías de programación de servidor: Servlets de Java, IDC/HTX de MS, CF, PHP.
- Multitud de navegadores, marcas y versiones.
- Mejora en las comunicaciones.
- Primeros portales generalistas con un masivo contenido multimedia.
- La web irrumpió en el mundo empresarial con fuerza.
- Año 1998, inicio de la llamada burbuja tecnológica de las empresas '.com'.

Sitios web: tercera generación

The screenshot shows the homepage of Terra Networks, dated Martes, 29 de febrero de 2000. The page features a large orange header bar with the Terra logo and navigation links for Buscador, Buscar en, Categorías, Avanzado, Añade tu web, and Ayuda. Below the header is a banner with three news snippets: "Haider dimite como presidente del Partido Liberal.", "Juegos on line 25 juegos para poner a prueba tu paciencia e ingenio.", and "Fútbol europeo: semana clave para seis equipos españoles.". The main content area is divided into several sections: "COMUNIDADES" (Chat, Correo web, Foros, Internet gratis, Mensajes a móviles, Postales, Tablones); "CANALES" (Actualidad, Arte, Bolsa de empleo, Buscador, Ciencia y tecnología, Cine, Comunicación, Cultura, Deportes, Economía y finanzas, Educación, Firmas de Terra, Gastronomía, Guías de ciudades, Hogar y familia, Informática, Internet, Jóvenes, Juegos, Medicina, Música, Sociedad, Terra Compras, Terramedia, Turismo); "HOY EN TERRA" (with links to Miss Internet, Elecciones 2000, Miss España, Arco 2000, and Páginas Amarillas); "TERRA COMPRAS" (with links to Tiendas, Productos, and various product categories like Informática, Escáneres, and Impresoras); "ESPECIALES" (with links to ELECCIONES 2000, Miss Internet, La Playa!, Arco 2000, and Páginas Amarillas); "TERRA RECOMIENDA" (with links to Alan Parsons, Sorteo, Educación, Basket europeo, and various news items); "EXCLUSIVOS" (with links to Invertia.com, Juegos Uproar, Canal Software, Promoción Univers, and Concursa con Kodak y Terra); and "ENCUESTA" (asking if there's a real chance for tax cuts). At the bottom, there are links for Ayuda, Página de inicio, Mi Terra, Infomail, Registro de usuarios, Publicidad, Sugerencias, Contactar, Leyenda, and a footer mentioning TERRA NETWORKS and various international presence. The footer also includes a copyright notice for Terra Networks, S.A. and a link to the Fundación Terra.

Sitios web: tercera generación



Sitios web: cuarta generación

- A partir del año 2002. Coincidiendo con la explosión de la burbuja tecnológica.
- Madurez del sector, diseño de webs más inteligente, minimalista al principio.
- Múltiples tecnologías de cliente: HTML, CSS, Javascript que junto con navegadores potentes permiten mejorar la interactividad.
- Múltiples tecnologías de servidor que permiten diseñar verdaderas aplicaciones en la web.

Sitios web: quinta generación

- **Aparición de la web 2.0. Web orientada al usuario, no a los contenidos.**
- Participación, contenidos aportados por los usuarios, comunidades virtuales, etc.
- RIA: Aplicaciones de Internet ricas (por interactivas).
- Web semántica, indexación más inteligente, buscadores más eficientes, aumento de las automatizaciones, PLN.
- Inicio de las redes sociales con pocos usuarios y servicios.

Sitios web: actualidad

- **Nuevas tendencias, web 3.0?**
- Web omnipresente: TV, radio, publicidad, teléfonos, etc.
- Redes sociales: aumento de su popularidad, implantación global.
- Nuevas formas de tener una web personal, empresarial, nuevas formas de relacionarse.
- Ubicuidad: tablets y smartphones: acceso a la web desde cualquier sitio, sin estar ligado a un PC.
- Boom de javascript, :-?

Sitios web: futuro

???

Sitios web: tendencias

- Ubicuidad TOTAL e 'inmediata' (Wifi, 5G)
- Video y multimedia, alta calidad, sin limitaciones.
- GIS, Geolocalización
- Realidad aumentada
- IoT: Internet of things (smart devices: relojes, teles, coches, electrodomésticos)
- IA & Semantics (personalización, data-mining)
- Webservices (M2M)
- Interoperabilidad entre plataformas, intercambio de información, sindicalización de contenidos, automatismos
- Servidores con contenidos dedicados (juegos, música, video, datos, etc.)
- Blockchain, bitcoins, NFTs
- Metaverso(s)

Sitios web: problemas

- Hay una corriente crítica, minoritaria, contra el servicio web y la red Internet por causas éticas.
- Incluso Tim Berners-Lee hizo unas declaraciones 'arrepintiéndose'
- Problemas que se dan:
 - Virus, Phising, timos, trampas, robos, suplantación de identidad
 - Hackeo (más servidores expuestos)
 - Contenidos 'basura', ilegales
 - Fake news, corrientes de opinión extremas, o paracientíficas, sobre todo ayudadas por las redes sociales y sus efectos virales.

Aplicaciones web

- Plataformas de ejecución de aplicaciones informáticas:
 - Aplicaciones de escritorio o autónomas, según el SO:
 - Windows
 - Unix
 - Apple
 - **Aplicaciones en entorno web**
 - Aplicaciones para dispositivos portables (Apps) para móviles, tablets, etc. Según el HW y SO:
 - Android
 - iOS
 - Windows Phone
- **Tendencias: desktop--, app++, web??**

Aplicaciones web

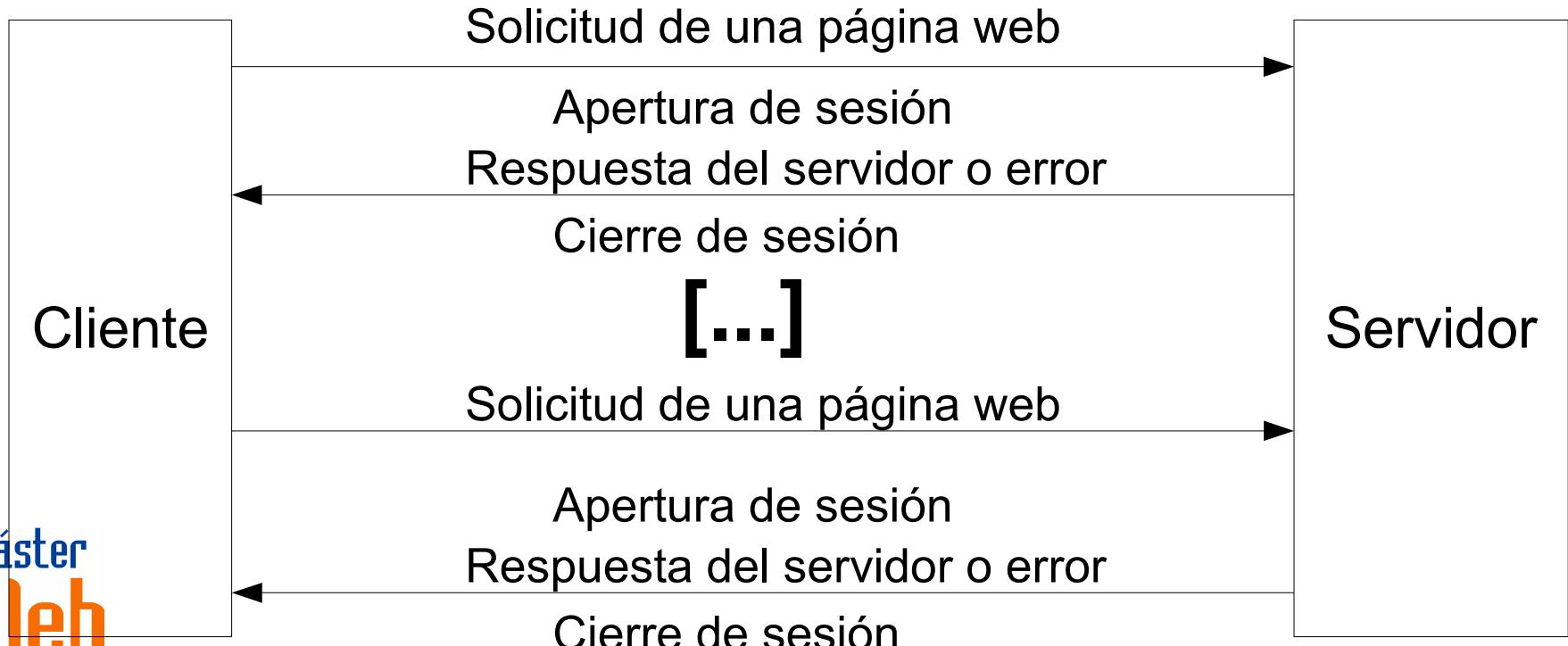
- Desde hace años muchos profesionales optan por desarrollar aplicaciones web en lugar de aplicaciones de escritorio.
- Razones:
 - Facilidad de acceso desde el cliente:
 - Solo hace falta un navegador web + una conexión.
 - Acceso desde cualquier sitio y desde cualquier dispositivo.
 - No es necesario un terminal muy potente, pues es el servidor quien realiza el trabajo.
 - Facilidad de desarrollo, instalación y actualización.
 - 'Multiplataforma'
 - Usuarios ya formados: la mayoría de la gente está familiarizada con internet, la web y los navegadores.
 - Tecnologías de comunicación fiables y rápidas.

Limitaciones del entorno

- Un sitio web tiene una arquitectura de aplicación basada en HTTP con sus retardos y posibles vulnerabilidades.
- Además, existe una limitación de controles o elementos en pantalla para interactuar con el usuario.
 - **Confinamiento en el HTML.**
 - Lo que no soporta el HTML cada uno 'se lo inventa', haciendo controles diferentes con la misma función.
 - **Solución:** uso de frameworks de cliente.
 - Inconsistencia de interfaz: una misma página puede parecer diferente o comportarse diferente según la plataforma, navegador o versión del cliente.

Funcionamiento de una aplicación web

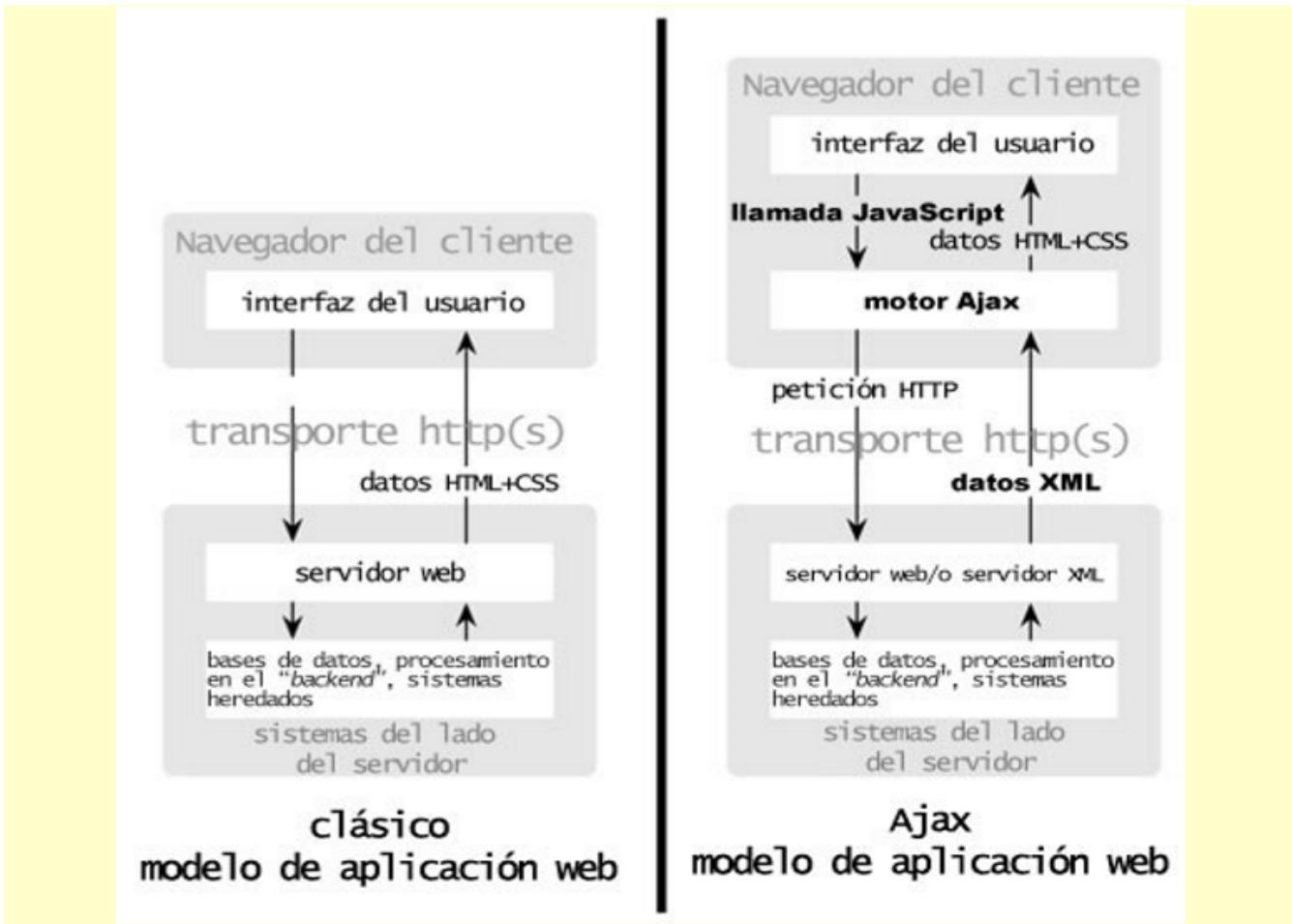
- Basado en el protocolo HTTP (que no mantiene la conexión entre el cliente y el servidor).
- Cada petición implica establecer la conexión, enviar los contenidos solicitados y cerrar la conexión.
 - Esto implica un retardo en la interacción del usuario.



RIA

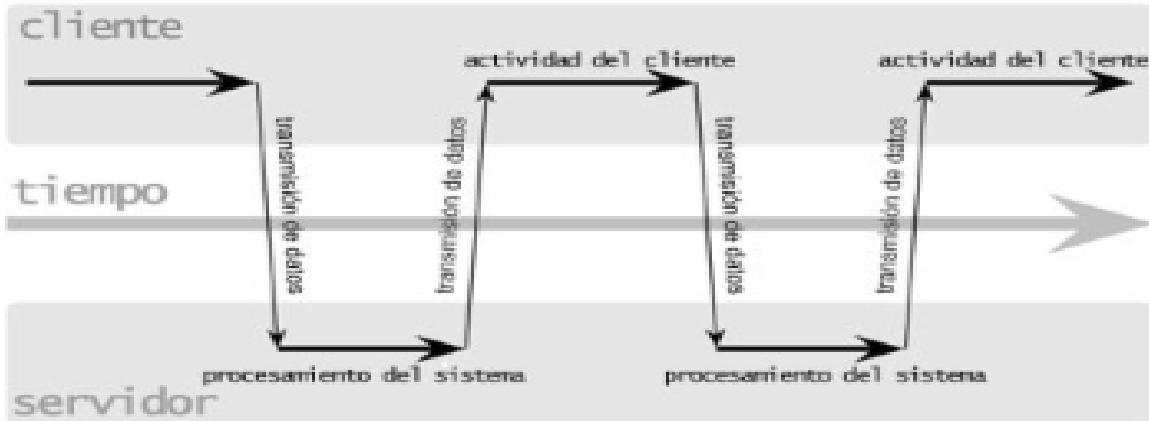
- Rich Internet Applications
- La aplicación web no envía una única petición para cargar la página, sino que envía micropeticiones en respuesta de las necesidades del usuario, cargando solo los datos que se necesitan.
- Cada micropetición se realiza de forma asíncrona, sin que la página deje de funcionar.
- Se pueden generar micropeticiones sin necesidad de que las desencadene el usuario para anticiparse a las necesidades de datos de éste y ofrecer una respuesta más rápida.

RIA

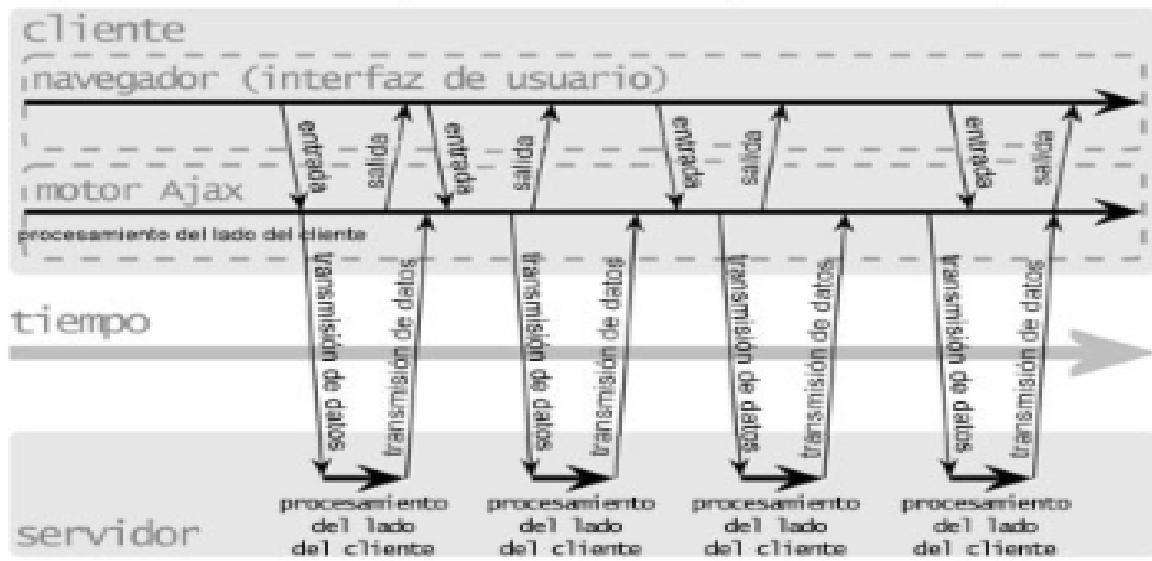


RIA

clásico modelo de aplicación web (sincrónica)



Ajax modelo de aplicación web (asincrónica)

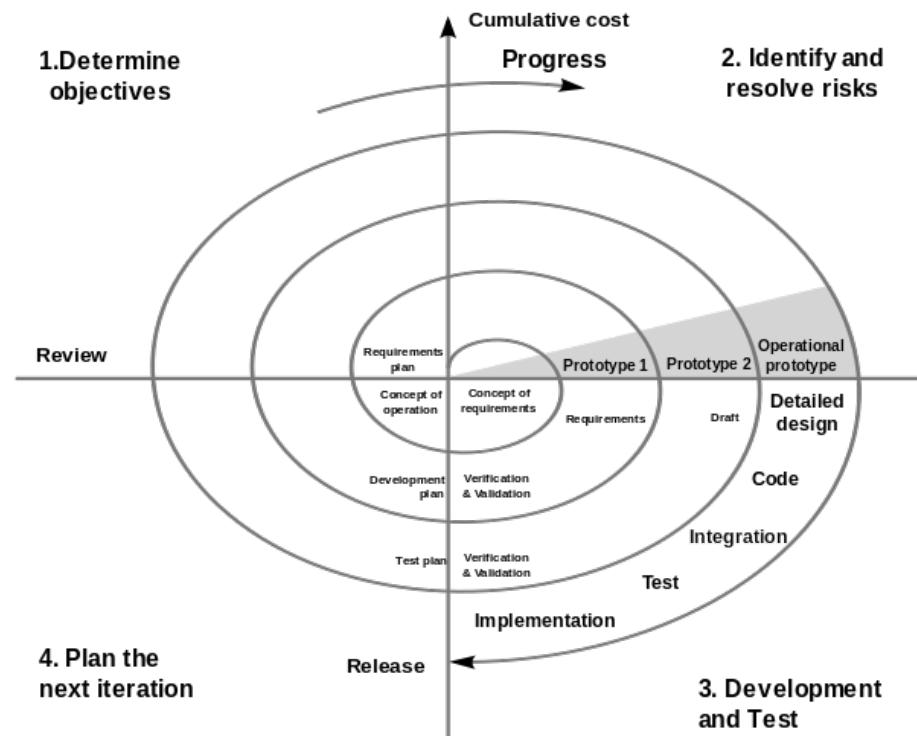
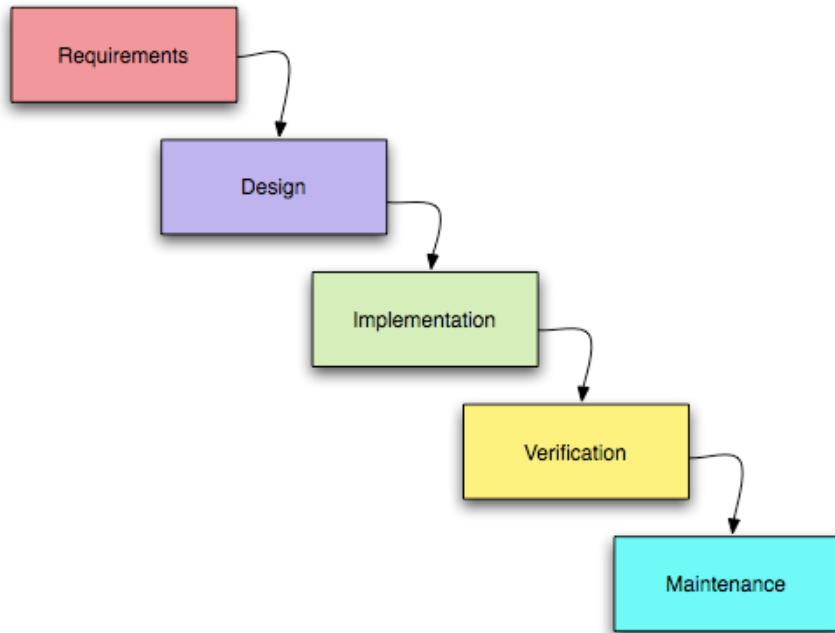


Ciclo de vida de un sitio web

- Existen varias metodologías.
- Ciclo de vida clásico del SW:
 - Análisis
 - Diseño
 - Implementación
 - Pruebas
 - Producción
 - Mantenimiento
- Ciclo de vida del SW por prototipado o espiral:
 - Análisis
 - (Re)Diseño (*)
 - Implementación
 - Pruebas
 - Demo del prototipo
 - Vuelta a (*) hasta OK
 - Producción
 - Mantenimiento
- Otras metodologías: Incremental, ágiles

Ciclo de vida de un sitio web

- Ciclo de vida clásico del SW
- Ciclo de vida del SW por prototipado o espiral:



Desarrollo de un sitio web

- Consiste en las tareas necesarias para desarrollar/implementar un sitio web
- En el desarrollo de un sitio web se han de abordar las siguientes tareas, entre otras:
 - Implementación de la parte de cliente
 - Implementación de la parte de servidor
 - Implementación de la base de datos: tablas y objetos necesarios
 - Diseño gráfico del sitio web (montaje de la página, infografía, tipografía, etc.)
 - Desarrollo del contenido de la web (textos, imágenes, videos, documentos, etc.)
 - Configuración de un servidor web y protección ante ataques.

Desarrollo de un sitio web: especialistas

- Web designer: diseñador web, tecnologías de cliente.
- Front-end developer: tecnologías de cliente, no diseño
- UI Designer: diseñador visual (bocetos, mockups)
- UX designer: user experience, usabilidad
- Art director
- Web developer: programador tecnologías de servidor.
- Content manager/strategist: responsable de contenidos.
- IT technician: hardware, redes, servidores.
- Dev Ops: Administradores de sistemas, deployment, testing, updates.
- Product manager: director del proyecto
- SEO Specialist

¿Qué es un Framework?

- Es una plataforma de software sobre la cual otro proyecto de software puede ser organizado y desarrollado.
- Puede incluir programas, bibliotecas de funciones y a veces un lenguaje interpretado propio.
- Condiciona al desarrollador a seguir una estructura y metodología determinada.
- Ventajas: facilitan el desarrollo de software permitiendo centrarse en los detalles del problema dejando los aspectos técnicos en segundo plano.

¿Qué es un Framework?

- Características:
 - Un framework toma el control de la ejecución, impone su forma de funcionar.
 - Un framework posee un comportamiento por defecto allá donde el programador no haya especificado nada.
 - Un framework es ampliable en cuanto a prestaciones así como internamente.
 - El código de un framework no debe ser modificado (no recomendado) salvo para ampliaciones.

Ejemplos de frameworks

- Ruby on rails (Ruby, MVC)
- Spring (Java, MVC)
- Express.js (Javascript+Node.js, API REST)
- Symfony (PHP, MVC)
- ASP.NET MVC (>Visual Studio 2010, C#, MVC)
- CodeIgniter (PHP, MVC)
- Django (Python, M-VM-M)
- Laravel (PHP, MVC)

Modelo Vista Controlador (MVC)

- Se trata de una arquitectura de software en la que se separan los datos de la interfaz y de la lógica de control.
 - Modelo: Representación de la información. Lógica de datos.
 - Vista: Interfaz de usuario. Front-end. Información que se envía al cliente y la interacción con éste.
 - Controlador: Lógica de control. Recoge las acciones del usuario (eventos) e invoca cambios en el modelo y también en la vista. Intermediario entre modelos y vistas.

Modelo Vista Controlador (MVC)

- MVC tiene en cuenta al usuario: todo se basa en sus acciones.
- Las acciones del usuario en la vista son recogidas por los controladores, entonces éstos realizan las acciones apropiadas contra el modelo y, en consecuencia, actualizan las vistas.
- La modularidad de MVC permite aprovechar una misma vista o controlador en diferentes módulos o aplicaciones.

Modelo Vista Controlador (MVC)

- Modelos:
 - Acceden a la capa de almacenamiento de datos.
 - Han de garantizar independencia de plataforma en la medida de lo posible.
 - Define las reglas de la lógica de datos, es decir, la funcionalidad del sistema, que operaciones se pueden hacer con los datos y que operaciones no están permitidas ni contempladas.

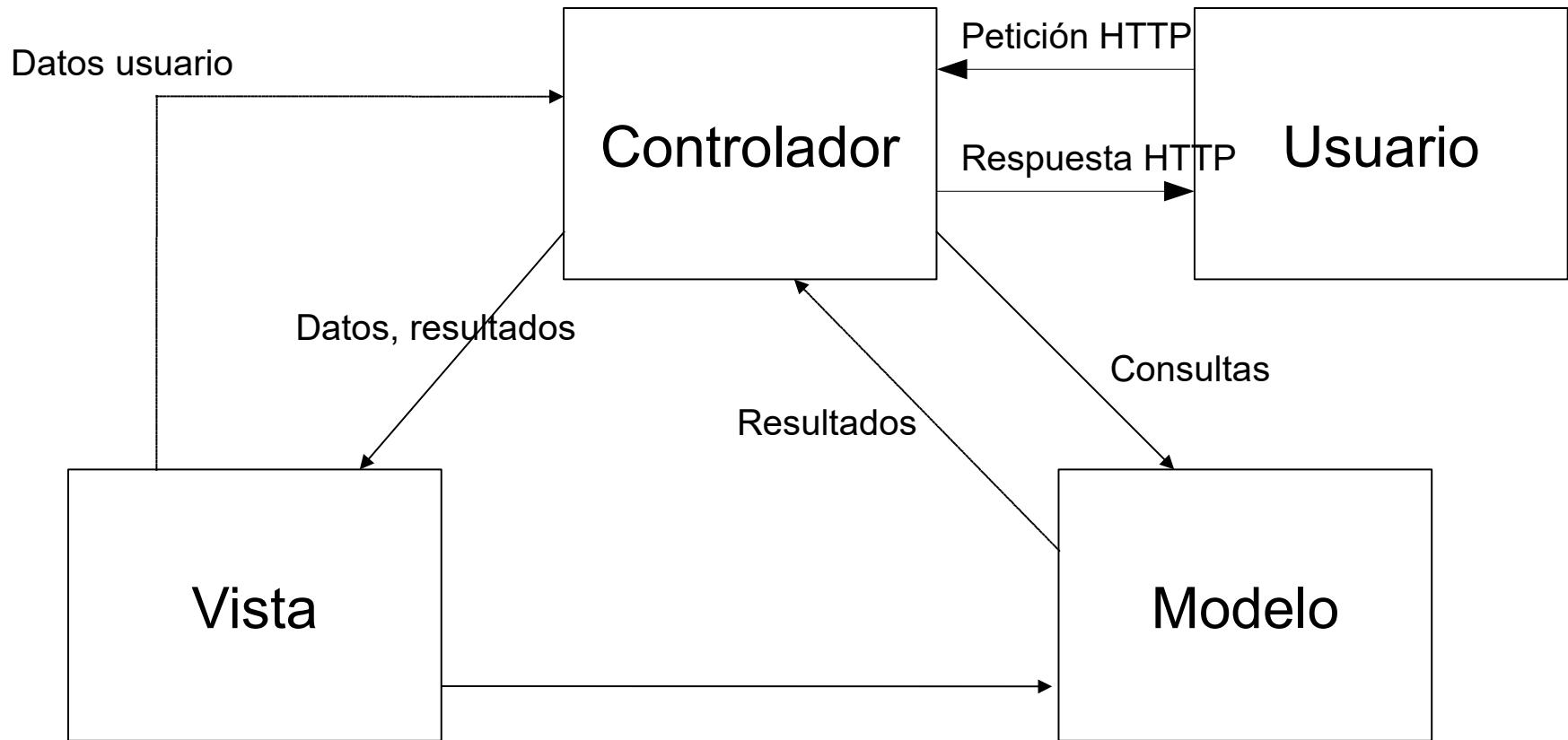
Modelo Vista Controlador (MVC)

- **Vistas:**
 - Reciben los datos del modelo y los muestran al usuario.
 - Están asociadas a un controlador y una acción del mismo.
- **Controladores:**
 - Reciben datos e interacción del usuario.
 - Contiene las reglas de la gestión de eventos. Si evento X -> Acción Y.
 - Las acciones pueden suponer peticiones a modelos (operar con los datos) e invocaciones de vistas (actualizar o mostrar datos)

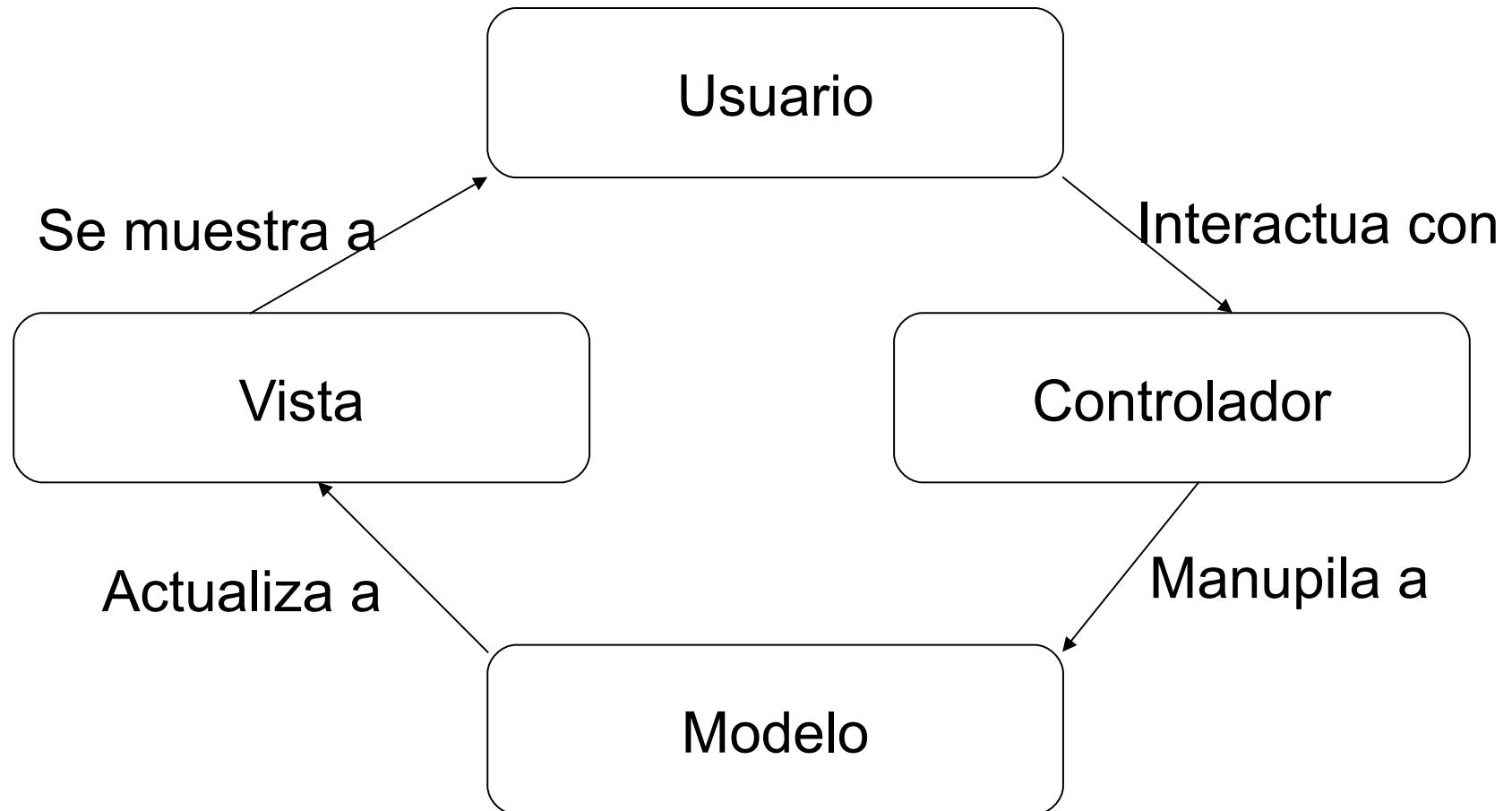
Modelo Vista Controlador (MVC)

- Flujo de trabajo:
 - El usuario interactua con el interfaz y provoca eventos o envia datos
 - El controlador recibe desde los componentes de la vista un evento e invoca la accion solicitada.
 - El controlador accede al modelo para consulta o actualización de datos.
 - El controlador envia los datos/resultados del modelo a la vista delegando en sus objetos la tarea de mostrar el nuevo interfaz reflejando los cambios provocados por la acción del usuario.
 - La vista o interfaz de usuario queda en espera de nuevos eventos y comienza el ciclo de nuevo.

Modelo Vista Controlador (MVC)



Modelo Vista Controlador (MVC)



Modelo Vista Controlador (MVC)

- Ventajas:
 - Clara separación de responsabilidades entre interfaz, lógica de negocio y de control.
 - Facilidad para la realización de pruebas unitarias de los componentes, así como de aplicar desarrollo guiado por pruebas (TDD).
 - Simplicidad en el desarrollo y mantenimiento de los sistemas.
 - Reutilización de los componentes.

Modelo Vista Controlador (MVC)

- Ventajas:
 - Facilidad para desarrollar prototipos rápidos.
 - Sencillez para crear distintas representaciones de los mismos datos.
 - Los sistemas son muy eficientes, y en consecuencia más escalables.

Modelo Vista Controlador (MVC)

- Inconvenientes:
 - Obligación de adaptarse a una estructura predefinida que puede incrementar la complejidad del proyecto.
 - Hay problemas cuya solución es más costosa respetando el patrón MVC.
 - Curva de aprendizaje costosa, sobre todo a desarrolladores acostumbrados a otros modelos, como Webforms.
 - La distribución de componentes obliga a crear y mantener un mayor número de ficheros.

Visual Studio

- IDE de microsoft, versión actual 2022.
 - Primera versión Visual Studio 97
 - Varias versiones y distribuciones
- Permite desarrollar aplicaciones gráficas de escritorio, para web, móviles, webservices, etc.
- Soporta varios lenguajes en el mismo entorno: C++, Visual Basic, C#, F++
- Permite la instalación de plugins para aumentar sus prestaciones: gestor de paquetes **Nuggets**

Visual Studio

- Características:

- Editor avanzado de código (con syntax highlighting, code completion, collapsing code blocks, etc.)
- Depurador
- Designer: forms, clases, webforms, data, etc.
- Object browser, properties editor
- Solution explorer, Data explorer, Server Explorer, Dotfuscator
- Web site administration tool
- Optimizadores de código: Resharper

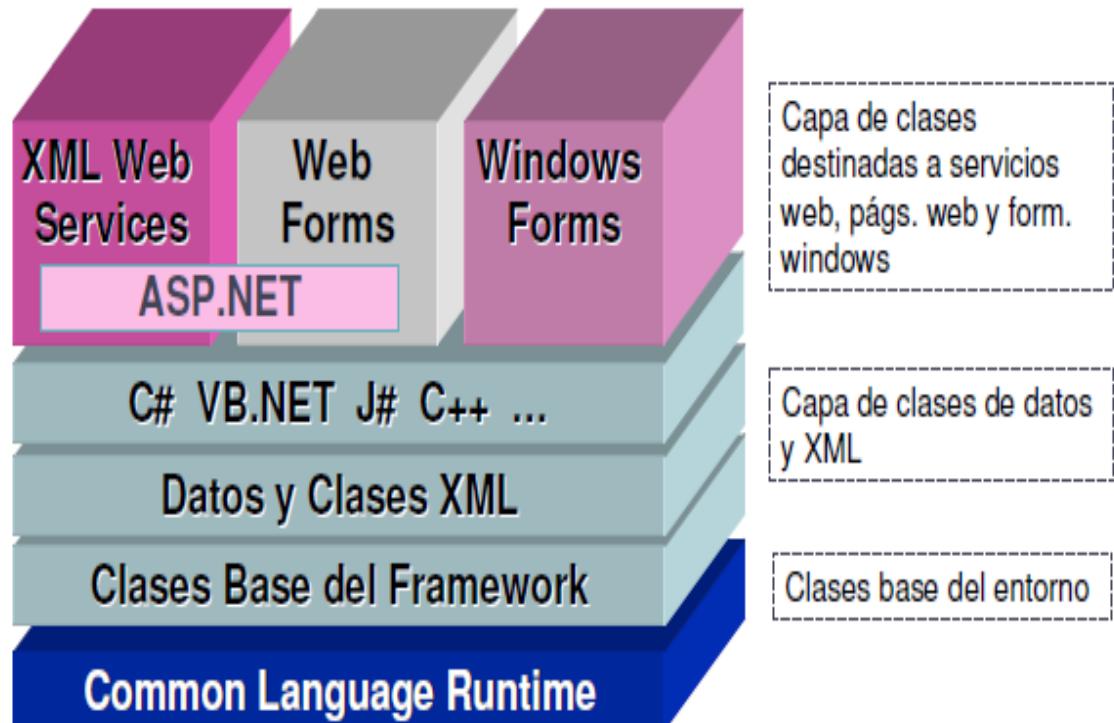
ASP.net

- Framework de desarrollo de aplicaciones web
 - Tecnología Server-side → páginas web dinámicas
 - Desarrollo de sitios web, aplicaciones web, servicios web.
 - Primera versión, enero 2002, sustituyendo a ASP
 - Última versión, abril 2018. Ver. 4.8
 - Sustituido por ASP.net CORE.

El framework .net



Framework de Microsoft .NET



Webforms

- Un webform es una mezcla entre un formulario de windows y una página HTML.
 - tiene apariencia y comportamiento de página web
 - Tiene controles de servidor que responden a eventos de usuario y con funcionalidad programable (winform)
- Como en un winform, se arrastran controles en su interior, se definen sus propiedades y se programan sus manejadores de eventos.
- Los usuarios interactuan con los elementos de la página web desde su navegador.

ASP.net webforms, características

- Un documento ASP.net o un web form consta de dos ficheros:
 - .aspx: contiene XHTML estático, etiquetas de servidor (Web controls, User controls) y ocasionalmente código de servidor (Entre llaves: <% ... %>)
 - .aspx.cs (.vb, ...): código de servidor asociado al web forms en (lenguaje C#, Visual Basic, etc.), fichero de 'code behind'.
- Uso de directivas, con <%@ Page %>
 - Una directiva es un comando especial que indica al compilador como procesa la página.

ASP.net webforms, características

- User controls (ascx)
 - encapsulación de un fragmento de código (tanto de cliente como de servidor)
- Custom controls (dll)
 - Como un user control, pero compilado en una DLL
- Gestión de estado: application, session, view state, server-side caching.
- Motor de plantillas o masterpages.

Application

- Variables globales del sitio web
- Accesibles desde cualquier página del sitio, cualquier usuario y cualquier momento.
- Se mantienen desde que el sitio web se pone en marcha (o desde que se instancian) hasta que el sitio web se detiene.
- Se pueden declarar, consultar o modificar desde cualquier lugar del aplicativo:
 - Application[“variable”]=valor;
- El lugar adecuado para declararlas es el fichero global.asax

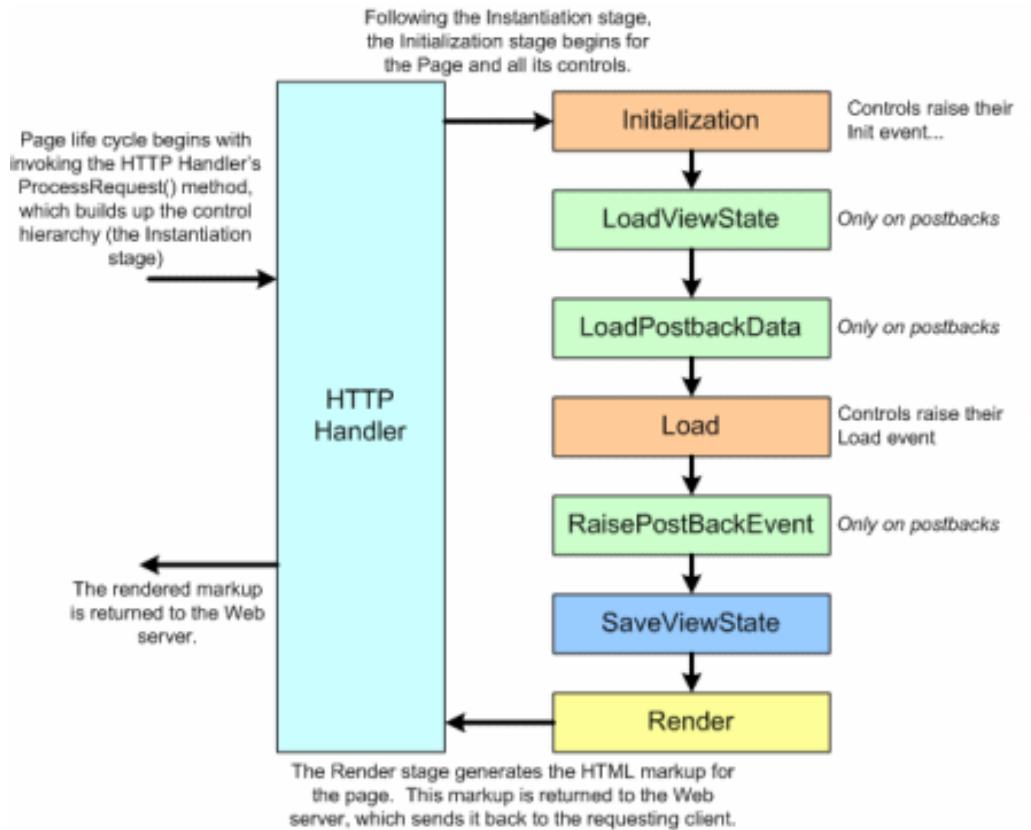
Session

- Variables globales para la sesión o visitante
- Accesibles desde cualquier página del sitio, pero únicas por usuario y con caducidad
- Se mantienen desde que el usuario accede al sitio web (o desde que se instancian) hasta que el usuario cierra la sesión o esta caduca por inactividad.
- Se pueden declarar, consultar o modificar de esta forma:
 - Session[“variable”]=valor;
- Se pueden declarar en el fichero global.asax: función session_start.

ViewState

- Viewstate se puede definir como el estado del ASPX entre diferentes PostBacks o peticiones.
- Unico para una sesión y ASPX dados en el tiempo.
- Se puede deshabilitar para todo el webform o para un control en particular.
- En el viewstate se almacena el estado de la página web entre diferentes llamadas o peticiones:
 - Se guardan los datos/contenido/propiedades de todos los controles de servidor.
 - Usa un campo hidden llamado '_VIEWSTATE' el cual puede ser relativamente grande en Kbs.
 - Cuidado con los controles añadidos dinámicamente

ViewState



- Artículo interesante:
 - <http://msdn.microsoft.com/en-us/library/ms972976.aspx>

ASP.net MVC

- Es un framework con arquitectura MVC para desarrollo de webs con ASP.NET de microsoft.
- Disponible a partir del Visual Studio 2010 como una alternativa (que no sustitución) a los Webforms.
- ASP.net 2.0 y 3.0 disponible solo a partir de Visual Studio 2010.
- ASP.net 4.0 disponible con Visual Studio 2012
- Un proyecto ASP.net MVC utiliza las siguientes tecnologías:
 - Servidor: C#, Razor (motor de vistas)
 - Cliente: Html, Javascript, CSS

ASP.net MVC

- Versiones actuales/finales: (noviembre 2018)
 - Entorno Visual Studio 2017
 - ASP.net MVC 5.2.7
- Versión más implantada (o estable)
 - Entorno Visual Studio 2017
 - ASP.net MVC 5
- Otros productos o componentes usados en conjunto frecuentemente:
 - Entity Framework 6
 - SQL server Express

Razor: motor de vistas

- Disponible a partir de ASP.net MVC3
- Permite incrustar código de servidor (en C# o Visual Basic) entre el código de cliente de una vista.
- Muy optimizado a la hora de compilar el código de servidor con el de cliente.
- Tiene una sintaxis ligera, no es necesario aprender un nuevo lenguaje.
- Los ficheros tienen extension .cshtml

Razor: motor de vistas

- Sintaxis:
 - Usa @ para incrustar variables:

```
<span>@ViewBag.datos</span>
```

- Sentencias de control:

```
@for (...) {  
}  
  
@if (...) {  
...@variable  
}
```

Razor: motor de vistas

- Sintaxis:
 - Bloques de código:

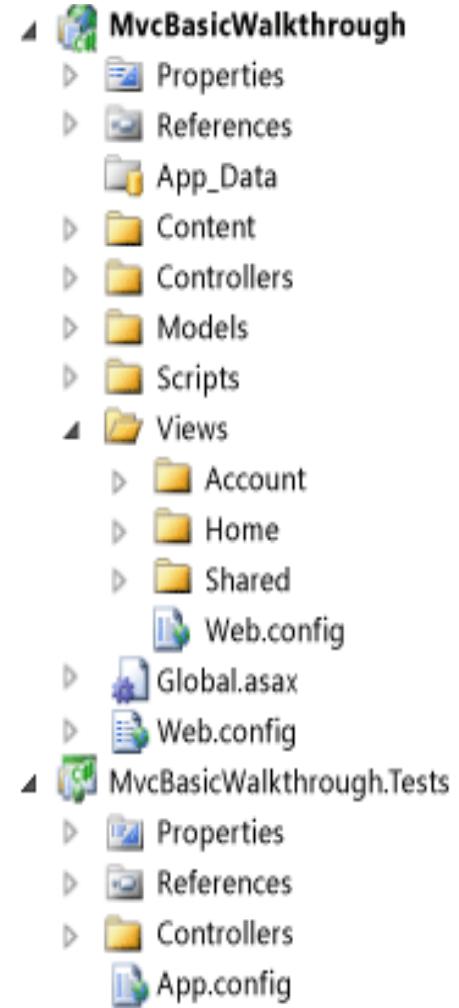
```
@{ ...  
  
    string uno="uno";  
  
    ... }
```

- Comentarios:

```
@* ...  
  
    Multilinea  
  
    ... *@  
  
// una sola linea
```

Estructura de un proyecto

- Estructura de un proyecto ASP.net MVC
 - App_Data: mismo rol que con Webforms
 - Content: contenidos del web.
 - CSS, imágenes, ficheros estáticos.
 - Controllers: controladores
 - Models: modelos de acceso a datos.
 - Scripts: ficheros de script Javascript, Jquery, etc.
 - Views: vistas. Una carpeta por controlador
 - Shared: vistas compartidas, plantillas.



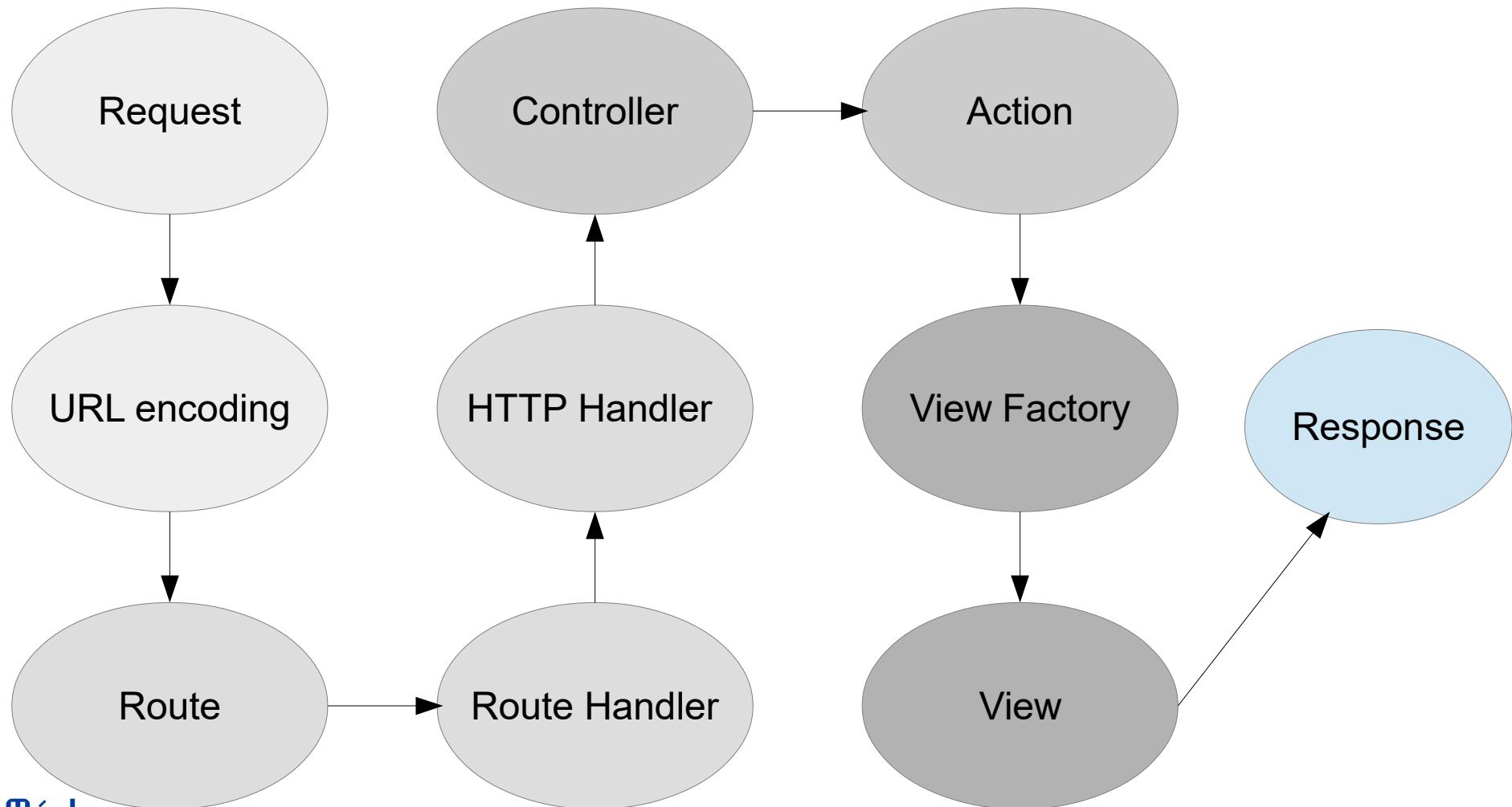
Ciclo de ejecución ASP.net MVC

- Fases de ejecución en un proyecto MVC:
 - Recepción de la petición (request)
 - El global.asax crea el RouteTable
 - Enrutado, se crea el objeto RequestContext.
 - Creación del manejador de la petición MVC.
 - Se crea el objeto MVCHandler y se le envia el RequestContext
 - Creación de una instancia del controlador
 - ...sigue...

Ciclo de ejecución ASP.net MVC II

- Fases de ejecución en un proyecto MVC:
 - ... tras crear la instancia del controlador...
 - Ejecución del controlador: invocación del método Execute.
 - Ejecutar la acción: se determina que acción se debe invocar y se invoca su método.
 - Ejecución del resultado:
 - Se envia la entrada del usuario a la acción.
 - La acción retorna un tipo de resultado:
 - ViewResult (renderiza una vista)
 - RedirectToRouteResult, RedirectResult, ContentResult, FileResult, EmptyResult, etc.

Ciclo de ejecución ASP.net MVC



Diferencias con Webforms

- No existe el postback, ni el viewstate, ni eventos.
- El diseñador visual deja de tener utilidad.
- No hay controles de servidor -> Helpers.
- No es necesario utilizar los archivos code-behind.
- El proceso de una petición es más simple.
- Control total sobre el código de cliente generado.
- Además de variables de sesión, permite otras modalidades con el ViewBag, ViewData, TempData

Diferencias con Webforms

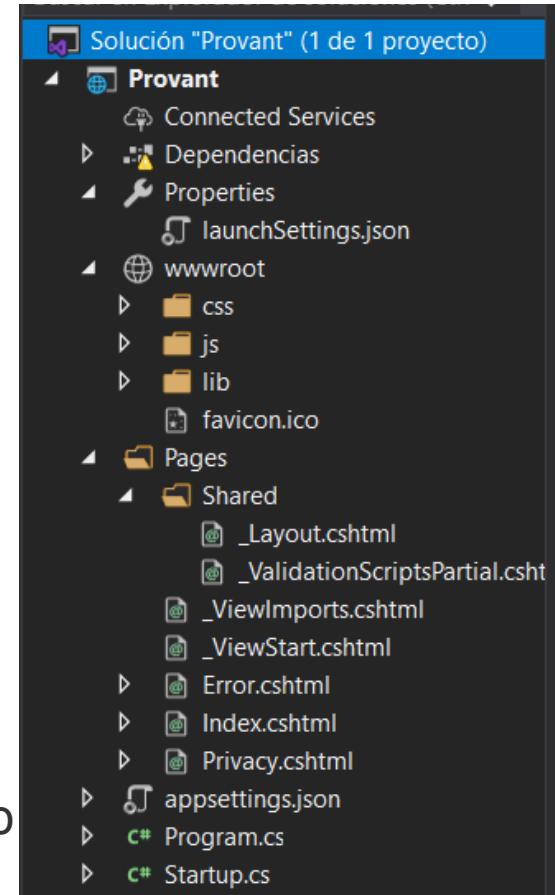
- Control sobre las URLs de acceso a nuestra aplicación.
- El framework es adaptable a nuestras preferencias.
 - P.E.: Cambiar Razor como motor de vistas por Spark.
- Se integra con Ajax de forma natural, sin artificios.
- No hay user controls para reutilizar código -> Partial Views.
- ASP.NET MVC framework ES software libre

ASP.net Core

- Framework de creación de sitios web open-source
- Refinamiento de ASP.net MVC, más rápido y compacto.
- Versión actual:
 - Noviembre de 2021
 - Entorno Visual Studio 2022 o superior.
 - Ver. 6.0.0
- Características más relevantes:
 - Modular, con el gestor de paquetes NuGet.
 - Unifica el desarrollo de UI web y web API
 - Multi-servidor y multiplataforma
 - Open-source
- Tipos de proyectos:
 - **ASP.net CORE con Razor Pages**
 - **ASP.net CORE con estructura MVC**

Estructura de un proyecto

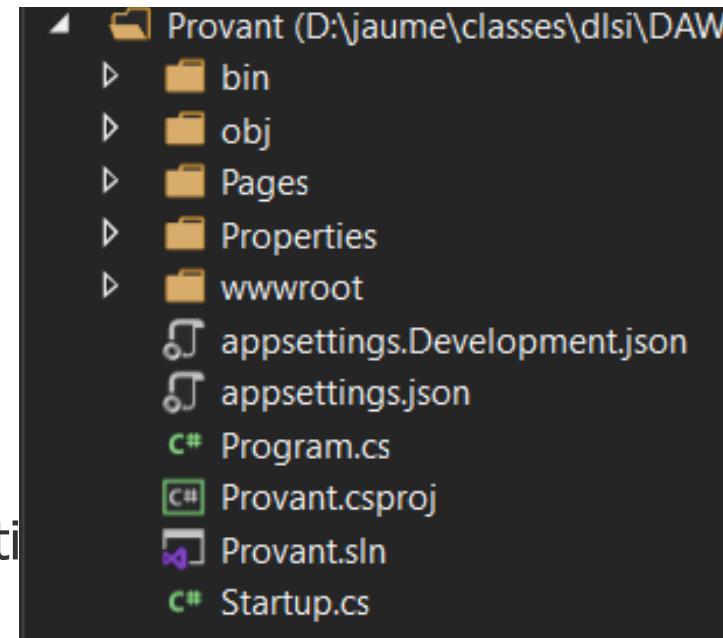
- Estructura de un **proyecto** ASP.net CORE con Razor Pages
 - Dependencies: paquetes nuget instalados en el proyecto
 - Properties:
 - LaunchSettings.json: perfiles de configuración para depurar código
 - Wwwwroot: directorio raíz, contenidos estáticos
 - Pages: páginas dinámicas del proyecto
 - Pages/shared: páginas/vistas compartidas
 - Ficheros:
 - Appsettings: datos globales de configuración
 - Program.cs: fichero inicial al ejecutar, con el método 'main()'
 - Startup.cs: equivalente al fichero 'global.asax'



Estructura de un proyecto

- Estructura de **carpetas** de un proyecto ASP.net CORE con Razor Pages

- Bin: Referencias, dlls.
- Obj: proyecto compilado
- Pages: páginas dinámicas del proyecto
- Properties: perfiles de configuración para depurar código
- Wwwwroot: directorio raíz, contenidos estáticos
- Ficheros:
 - Appsettings
 - Program.cs
 - Startup.cs
- .csproj, .sln: el fichero del proyecto y de la solución

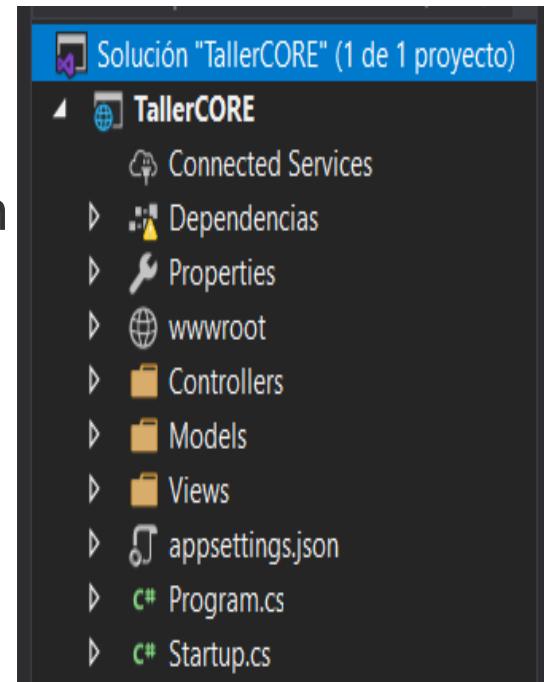


ASP.net CORE Razor Pages

- Con ASP.net CORE cada página es un ViewModel cuyo código se divide en dos ficheros:
 - Page.cshtml: contiene el código en Razor de la vista.
 - La directiva @page permite que la página se convierta en una URL invocable, una acción.
 - Page.cshtml.cs: contiene el código en C# del modelo (view-model), en forma de clase que hereda de la clase del sistema *PageModel*.
- Por convenio, 'index.cshtml' es el nombre para las rutas por defecto:
 - <https://servidor/proyecto/> equivale a <https://servidor/proyecto/index>

Estructura de un proyecto

- Estructura de un proyecto ASP.net CORE con estructura MVC
 - Dependencies: paquetes nuget del proyecto
 - Properties:
 - LaunchSettings.json: perfiles de configuración para depurar
 - Wwwwroot: directorio raíz, contenidos estáticos
 - Controllers: controladores del proyecto
 - Models: modelos/servicios del proyecto
 - Views: Vistas del proyecto
 - Views/shared: vistas compartidas
 - Ficheros:
 - Appsettings: datos globales de configuración
 - Program.cs: fichero inicial al ejecutar, con el método 'main()'
 - Startup.cs: equivalente al fichero 'global.asax'



Referencias

- <http://www.asp.net/mvc>
- <http://msdn.microsoft.com/en-us/library/dd381412%28v=vs.98%29.aspx>
- <http://www.variablenotfound.com/2010/05/aspnet-mvc-2-quince-cuestiones-que.html>
- <http://msdn.microsoft.com/en-us/library/ms978748.aspx>
- <http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html>
- <https://www.asp.net/mvc/overview/getting-started/recommended-resources-for-mvc>
- <https://www.tutorialsteacher.com/core/aspnet-core-application-project-structure>
- <https://docs.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-6.0&tabs=visual-studio>