**Resumen Ampliado: Plataformas de Programación Web en Entorno Servidor – Aplicaciones XAMPP**

**1. Funcionamiento de las Páginas Web**

Cuando una página web se descarga desde Internet, el navegador interpreta su contenido (HTML o XHTML) y muestra la información en pantalla.  
Los **estilos** se definen mediante un archivo CSS, que se descarga junto con la página, y que indica al navegador cómo debe presentarse (colores, fuentes, alineación, etc.).

El proceso de descarga de una página sigue un esquema **cliente-servidor**:

1. El navegador (cliente) solicita una página al servidor.
2. El servidor busca el archivo solicitado.
3. Si lo encuentra, lo envía al cliente.
4. El navegador lo muestra en pantalla.

**1.1 Páginas Estáticas y Dinámicas**

**Páginas Estáticas**

* Son archivos fijos que no cambian a menos que el programador los edite.
* Utilizan HTML/XHTML y CSS.
* No requieren programación avanzada ni bases de datos.
* Ventajas:
  + Simples, rápidas, seguras y fáciles de alojar.
  + Su contenido es fácilmente indexable por buscadores como Google.
  + Pueden almacenarse en medios físicos y abrirse sin conexión.
* Desventajas:
  + No pueden personalizar contenido.
  + Su actualización es manual, lo que dificulta el mantenimiento.

**Páginas Dinámicas**

* Generan contenido variable en función de distintos factores: usuario, navegador, idioma, etc.
* Requieren la ejecución de un **programa en el servidor** (PHP, ASP, JSP, etc.) que genera el HTML final.
* Este proceso incluye:
  1. El cliente solicita una página dinámica.
  2. El servidor ejecuta un programa (por ejemplo, PHP).
  3. Si es necesario, el servidor accede a una base de datos.
  4. Se genera una página HTML personalizada.
  5. El resultado se envía al navegador.

Ejemplo: cuando accedes a tu cuenta de correo (Gmail, Outlook…), el servidor genera una página diferente para cada usuario.

**Ventajas y desventajas**

* Ventajas: personalización, interactividad, integración con bases de datos.
* Desventajas: requieren programación, mantenimiento de software adicional y recursos del servidor.

**1.2 Aplicaciones Web**

Las **aplicaciones web** son programas que se ejecutan en un servidor y se acceden desde un navegador (por ejemplo, Google Docs o Trello).

**Ventajas:**

* No necesitan instalación en el equipo del usuario.
* Pueden usarse desde cualquier dispositivo con Internet.
* Su mantenimiento y actualización son centralizados.

**Desventajas:**

* Requieren conexión constante.
* El rendimiento depende de la red y del servidor.
* Limitaciones gráficas frente a las aplicaciones de escritorio.

Estructuralmente se dividen en:

* **Front-end:** interfaz visible para el usuario.
* **Back-end:** gestión de datos, usuarios y lógica interna.

**1.3 Ejecución de Código en Cliente y Servidor**

* **Código en el cliente:** ejecutado por el navegador (JavaScript).
* **Código en el servidor:** ejecutado antes de enviar la página (PHP, JSP, ASP.Net).

Ambos se complementan:

* El servidor obtiene datos y genera HTML.
* El cliente realiza validaciones, animaciones o consultas asincrónicas.

Con **AJAX**, JavaScript puede comunicarse con el servidor sin recargar la página, mejorando la fluidez de las aplicaciones modernas.

**2. Tecnologías del Lado del Servidor**

**2.1 Componentes Principales**

* **Servidor Web:** gestiona peticiones HTTP (Apache, Nginx, IIS).
* **Módulo de ejecución:** interpreta el código dinámico (PHP, Java, ASP.Net).
* **Servidor de bases de datos:** almacena información (MySQL, MariaDB, PostgreSQL).

Las aplicaciones suelen estructurarse en **capas**:

1. **Presentación:** interfaz de usuario.
2. **Lógica:** procesamiento de datos.
3. **Acceso a datos:** comunicación con la base de datos.

**2.2 Principales Arquitecturas**

| **Arquitectura** | **Componentes** | **Características** |
| --- | --- | --- |
| **Java EE** | Java + Servlets/JSP | Robusta, modular, multiplataforma. Ideal para proyectos grandes. |
| **AMP (LAMP/WAMP/MAMP)** | Apache + MySQL + PHP | De código abierto, sencilla de aprender, muy popular. |
| **CGI/Perl** | Scripts ejecutables | Antigua y lenta, pero flexible. |
| **ASP.Net** | IIS + C# / VB.Net | Plataforma comercial de Microsoft, entorno integrado (Visual Studio). |
| **Node.js** | JavaScript + V8 | Moderna, rápida y no bloqueante. Ideal para apps en tiempo real. |

**2.2.1 Selección de una arquitectura de programación web**

Antes de empezar un proyecto, deben considerarse factores como:

* Tamaño y complejidad del sistema.
* Lenguajes de programación que el equipo domina.
* Si se usarán herramientas **open source o propietarias**.
* Coste de licencias.
* Tipo de despliegue (servidor propio o externo).
* Posible trabajo en equipo y uso de sistemas de control de versiones.

El análisis de estos puntos permitirá elegir la arquitectura más adecuada (por ejemplo, **AMP** para proyectos educativos o **Java EE** para soluciones empresariales).

**2.3 Integración con el Servidor Web**

La comunicación entre navegador y servidor se realiza a través del protocolo **HTTP**, mediante peticiones y respuestas.  
Cada arquitectura define su propia forma de integrarse con el servidor web:

**CGI**

* Define un estándar para que el servidor delegue en un programa externo la generación del contenido.
* Inconveniente: **crea un nuevo proceso por cada petición**, lo que aumenta el consumo de recursos.
* Soluciones: **FastCGI** (procesos persistentes) o módulos específicos (como *mod\_perl* para Apache).

**PHP**

* Normalmente se integra mediante el módulo **mod\_php** en Apache, que ejecuta el código directamente sin crear procesos nuevos.
* También puede usarse mediante CGI, aunque es menos eficiente.

**Java EE**

* Requiere **servidores de aplicaciones** o **contenedores de servlets**.
* **Servidores de aplicaciones** implementan toda la especificación Java EE (ej. IBM WebSphere, BEA WebLogic, JBoss, Geronimo, GlassFish).
* **Contenedores de servlets** como **Apache Tomcat** soportan servlets y JSP, y pueden incluir su propio servidor HTTP o integrarse con Apache.

**ASP.Net**

* Se ejecuta en **IIS**, que integra módulos para procesar las páginas ASP/ASP.Net.
* Incluye herramientas de administración y despliegue dentro del propio servidor.

**3. Lenguajes de Programación Web**

Existen tres grandes formas de ejecución del código del lado del servidor:

| **Tipo** | **Ejecución** | **Ejemplos** | **Características** |
| --- | --- | --- | --- |
| **Guiones (Scripting)** | Interpretados línea a línea | PHP, Perl, Python, ASP | Fáciles de editar y mantener; menor rendimiento. |
| **Compilados Nativos** | Traducidos a binario dependiente del sistema | C, C++ | Muy rápidos pero poco portables. |
| **Compilados Intermedios** | Generan código portable (bytecode) | Java, C# | Buen equilibrio entre velocidad y portabilidad. |