`java.awt.Desktop`:

Cuando necesitamos abrir un navegador para cargar una página Web desde una aplicación Java podemos utilizar los métodos que nos ofrece esta clase dentro del paquete de utilidades gráficas de Java.

La clase `Desktop` permite que una aplicación Java pueda arrancar las aplicaciones registrada en la *máquina nativa* que utilicemos para acceder a un URI o abrir un archivo. Esta clase y lso servicios que proporciona a las aplicaciones está disponible desde la versión 1.6 de Java.

Las operaciones que ofrece `Desktop` son las siguientes:
-arrancar el navegador seleccionado por defecto por el usuario para mostrar un URI dado
-arrancar el cliente de correo del usuario por defecto con un argumento opcional del tipo `mailto` URI
-arrancar una aplicación instalada en la máquina para abrir, editar o imprimir un archivo dado.

Desktop proporciona los métodos específicos para realizar estas operaciones. Los métodos buscan la aplicación asociada a cada uno que se encuentre instalada en la máquina local, y la arrancarán para que procese un archivo o un URI. Si no existe tal aplicación o si la aplicación fallara al iniciarse, se levantará una excepción.

Toda aplicación tiene asociado un tipo de archivo o un URI registrado; por ejemplo, la extensión de archivo "sxi" está normalmente asociada a las aplicaciones de StarOffice. El mecanismo de asociar, acceder y arrancar una aplicación determinada dependerá de la plataforma (sistema operativo, principalmente) que nos esté dando servicio.

Las operaciones a las que nos referimos anteriormente son un tipo de acción que viene representado por la clase `Desktop.Action`.

Cuando se invoca alguna acción y la aplicación asociada se consigue ejecutar, ésta lo será sobre el sistema operativo en el cuál se arrancó Java.

void	`browse(URI uri)` Arranca el navegador por defecto y muestra a lo que se liga el URI del argumento.
void	`edit(File archivo)` Arranca el editor asociado y abre el archivo argumento para editarlo.
static Desktop	`getDesktop()` Devuelve la instancia de `Desktop` correspondiente al contexto del navegador por defecto.
static boolean	`isDesktopSupported()` Comprueba si la clase `Desktop` y sus operaciones son asumidas en nuestra plataforma actual.
boolean	`isSupported(Desktop.Action accion)` Comprueba si la acción del argumento es soportada en nuestra plataforma actual.
void	`mail()` Arranca la ventana para crear correos del cliente de correo por defecto que tengamos definido.
void	`mail(URI mailtoURI)` Arranca la ventana para crear correos del cliente de correo por defecto que tengamos definido, rellenando los campos de mensaje especificados por una orden `mailto: URI`.
void	`open(File archivo)` Arranca la aplicación asociada para abrir el archivo.
void	`print(File archivo)` Imprime un archivo con la rutina de impresión nativa de nuestra plataforma, utilizando la orden de impresión de la aplicación asociada al tipo de archivo.

Y también define una clase estática anidada, que representa un tipo de acción:

static class	Desktop.Action

-Utilización de clase Desktop en la práctica:

Se puede utilizar `Runtime` para crearse una versión multiplataforma (Windows, Linux) de un sencillo programa de demostración que abrirá el URL que se pasa como argumento al método browse, utilizando para ello el navegador que tengamos definido por defecto:

```
package browser;
import java.awt.Desktop;
import java.net.URI;

public class Aplicacion {
        public static void main(String[] args) throws Exception {
               String url = "http://localhost:8080/holamundo/home.jsf";

               if (Desktop.isDesktopSupported()) {
                   // Windows
                   Desktop.getDesktop().browse(new URI(url));
               } else {
                   // Ubuntu
                   Runtime runtime = Runtime.getRuntime();
                   runtime.exec("/usr/bin/firefox -new-window " + url);
```

```
                }
            }
}
```

Después de compilarlo y ejecutarlo dentro del IDE Eclipse, se abrirá, p.e., el IExplorer y obtendremos lo siguiente en pantalla: