El futuro digital es de todos

MinTIC

Unidad 4

Universidad Tecnológica de Pereira

MisiónTIC2022

El futuro digital es de todos

MinTIC

Tema 9 - Debugger

Universidad Tecnológica de Pereira

MisionTIC2022

Cómo depurar tu app con Android Studio

Android Studio proporciona un depurador que te permite realizar las siguientes acciones y más:

- Seleccionar un dispositivo en el cual depurarás tu app
- Establecer interrupciones en tu código Java, Kotlin y C/C++
- Examinar variables y evaluar expresiones en el tiempo de ejecución.

Se incluirán instrucciones para las operaciones básicas del depurador. Para obtener más documentación, también puedes consultar los documentos de depuración de IDEA de IntelliJ.

Cómo habilitar la depuración

Habilita la depuración en tu dispositivo:

Si estás usando el emulador, esta opción estará activada de forma predeterminada. Sin embargo, en el caso de un dispositivo conectado, deberás habilitar la depuración en las opciones para desarrolladores del dispositivo.

Ejecuta una variante de compilación depurable:

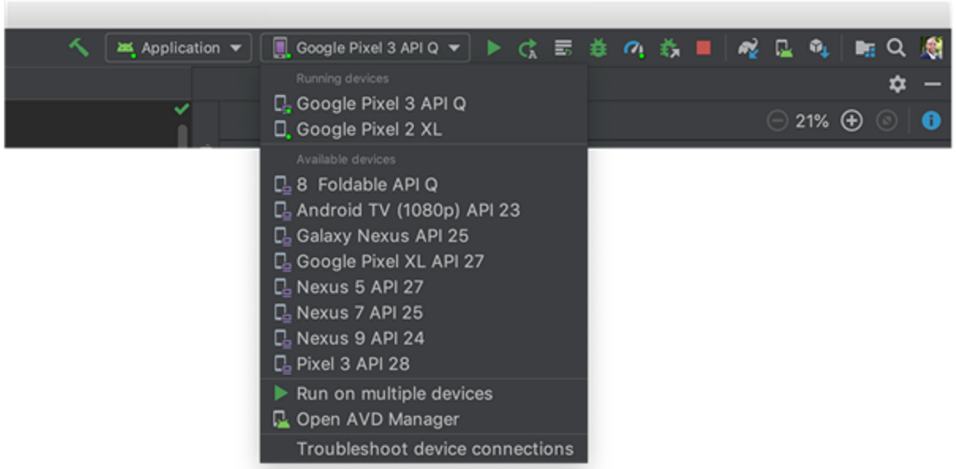
Debes usar una variante de compilación que incluye debuggable true en la configuración de compilación. Por lo general, simplemente puedes seleccionar la variante "debug" predeterminada que se incluye en cada proyecto de Android Studio (aunque no esté visible en el archivo build.gradle). Sin embargo, si defines nuevos tipos de compilación que deberían ser depurables, debes agregar "debuggable true" al tipo de compilación de la siguiente

Cómo iniciar la depuración

Puedes iniciar una sesión de depuración de la siguiente manera:

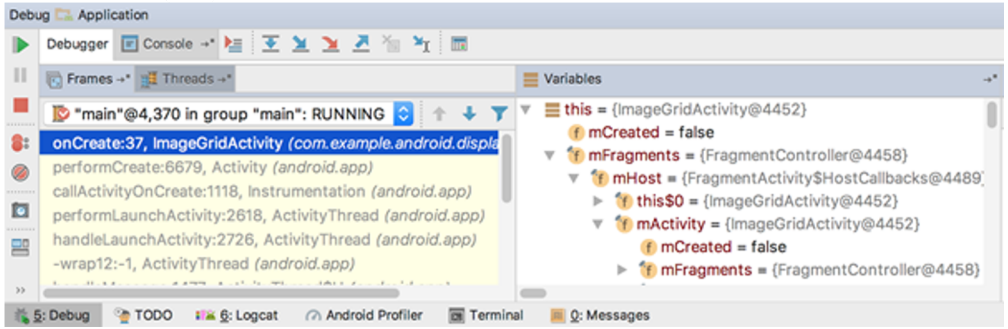
1. Establece algunas interrupciones en el código de la app.
2. En la barra de herramientas, selecciona el dispositivo en el que depurará la app en el menú desplegable del dispositivo de destino

   Si no tienes ningún dispositivo configurado, debes conectar uno mediante USB o crear un AVD para usar Android Emulator.



Cómo iniciar la depuración

3. En la barra de herramientas, haz clic en **Debug**  .
   Si se muestra el diálogo "switch from Run to Debug", significa que tu app ya se está ejecutando en el dispositivo y se reiniciará para comenzar con la depuración. Si prefieres conservar la misma instancia de la app en ejecución, haz clic en **Cancel Debug** y, en su lugar, adjunta el depurador a la app en ejecución.
   Android Studio compila un APK, lo firma con una clave de depuración, lo instala en el dispositivo seleccionado y lo ejecuta. Si agregas código C y C++ a tu proyecto, Android Studio también ejecutará el depurador LLDB en la ventana Debug para depurar tu código nativo.
4. Si la ventana **Debug** no está abierta, seleccione **View > Tool Windows > Debug** (o haz clic en Debug en la barra de ventanas de herramientas) y, luego, haz clic en la pestaña Debugger



Cómo adjuntar el depurador a una app en ejecución

Si tu app ya se está ejecutando en el dispositivo, puedes comenzar con la depuración sin reiniciarla de la siguiente manera:

1. Haz clic en **Attach debugger to Android process** 
2. En el diálogo **Choose Process**, selecciona el proceso al que desees adjuntar el depurador.
   Si estás usando un emulador o un dispositivo con derechos de administrador, puedes marcar la opción **Show all processes** para ver todos los procesos.
   En el menú desplegable **Use Android Debugger Settings**, puedes seleccionar una configuración de ejecución y depuración existente. (En proyectos con código C y C++, esto te permite reutilizar los directorios de símbolos y los comandos de arranque y posconexión de LLDB de una configuración existente). Si no tienes una configuración de ejecución y depuración, selecciona **Create New**. De esta forma, se habilita el menú desplegable **Debug Type**, donde puedes seleccionar un tipo de depuración diferente. De forma predeterminada, Android Studio usa el tipo de depuración **Auto** para seleccionar la mejor opción de depuración en función de si tus proyectos incluyen código Java o C/C++.
3. Haz clic en **OK**.
   Aparecerá la ventana **Debug**.

Cómo usar el registro del sistema

El registro del sistema **muestra mensajes mientras depuras tu app**, que incluyen información sobre apps que se ejecutan en el dispositivo. Si deseas usar el registro del sistema para depurar tu app, **asegúrate de** que tu código ***escriba mensajes de registro y también imprima el seguimiento de pila*** para las excepciones mientras tu app se encuentre en etapa de desarrollo.

Cómo escribir mensajes de registro en tu código

Para escribir mensajes de registro en tu código, usa la clase Log. Estos mensajes te ayudan a comprender el flujo de ejecución mediante la recopilación de los resultados de depuración del sistema mientras interactúas con tu app. Además, pueden indicar cuál es la parte de tu app que falló. Para obtener más información sobre el registro, consulta Cómo escribir y ver registros.

Cómo escribir mensajes de registro en tu código

```
import android.util.Log;
...
public class MyActivity extends Activity {
  private static final String TAG = MyActivity.class.getSimpleName();
    ...
    @Override
    public void onCreate(Bundle savedInstanceState) {
       ...
       if (savedInstanceState != null) {
            Log.d(TAG, "onCreate() Restoring previous state");
            /* restore state */
        } else {
            Log.d(TAG, "onCreate() No saved state available");
            /* initialize app */
        }
    }
}
```

Durante el desarrollo, tu código también puede detectar excepciones y escribir el seguimiento de pila en el registro del sistema:

```
void someOtherMethod() {
    try {
        ...
    } catch (SomeException e) {
        Log.d(TAG, "someOtherMethod()", e);
    }
}
```

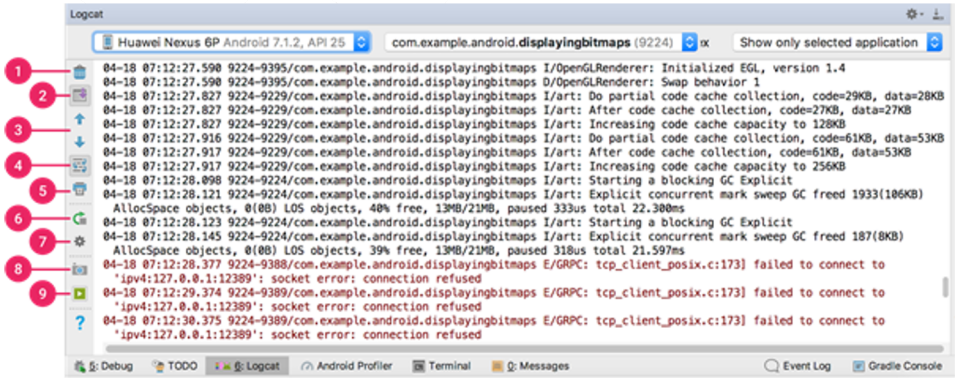
Cómo ver el registro del sistema

Puedes ver y filtrar la depuración y otros mensajes del sistema en la ventana **Logcat**. Por ejemplo, puedes ver mensajes cuando se produce la recolección de elementos no utilizados, o bien mensajes que agregas a tu app, con la clase Log.

Para usar Logcat:

1. Debes iniciar la depuración
2. Seleccionar la pestaña Logcat en la barra de herramientas inferior.

Se verá así:



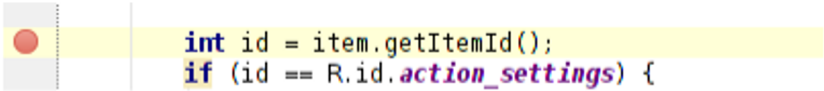
Para obtener una descripción de Logcat y sus opciones de filtrado, consulta Cómo escribir y ver registros con Logcat.

Cómo trabajar con interrupciones

- Android Studio admite varios tipos de interrupciones que activan diferentes acciones de depuración.
- El tipo más común es una interrupción de línea que detiene la ejecución de tu app en una línea de código especificada.
- Mientras la app está detenida, puedes examinar variables, evaluar expresiones y, luego, continuar la ejecución línea por línea para determinar las causas de los errores en el tiempo de ejecución.

Para agregar una interrupción de línea, haz lo siguiente:

1. Localiza la línea de código en la que desees detener la ejecución, luego haz clic en el margen izquierdo de esta o coloca el símbolo de intercalación en la línea y presiona Control + F8 (en Mac, Command + F8).
2. Si tu app ya se está ejecutando, no necesitas actualizarla para agregar la interrupción. Simplemente, haz clic en Attach debugger to Android process . De lo contrario, inicia la depuración haciendo clic en Debug .



Cuando estableces una interrupción, aparece un punto rojo junto a la línea.

Cuando la ejecución del código llega a la interrupción, Android Studio detiene la ejecución de tu app. Luego, puedes usar las herramientas de la pestaña Debugger para identificar el estado de la app:

Para examinar el árbol de objetos de una variable, expandelo en la vista **Variables**. Si no ves la vista **Variables**, haz clic en **Restore Variables View**

Para avanzar a la primera línea dentro de una llamada a un método, haz clic en **Step Into**.

Para evaluar una expresión en el punto de ejecución actual, haz clic en **Evaluate Expression**

Para avanzar a la siguiente línea fuera del método actual, haz clic en **Step Out**

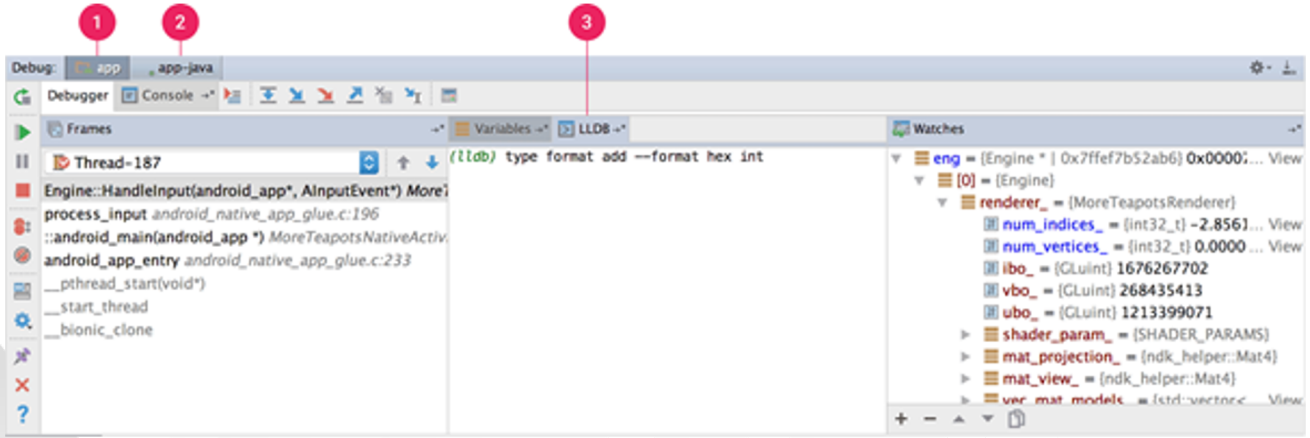
Para examinar el árbol de objetos de una variable, expándelo en la vista Variables. Si no ves la vista Variables, haz clic en Restore Variables View

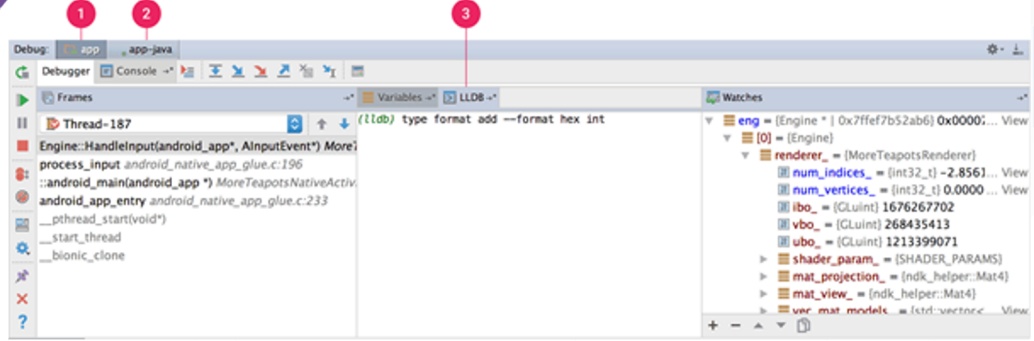
Para continuar ejecutando la app normalmente, haz clic en **Resume Program**

Cómo trabajar con interrupciones

Si tu proyecto usa código nativo, de forma predeterminada, el tipo de depuración Auto adjunta el depurador Java y LLDB a tu app como dos procesos independientes, de manera que puedas realizar el cambio entre la inspección de interrupciones de Java y C/C++ sin tener que reiniciar tu app ni cambiar la configuración.

Cuando Android Studio implementa tu app en el dispositivo de destino, se abre la ventana "Debug" con una pestaña o la vista de la sesión de depuración para cada proceso del depurador, como se muestra aquí





1. Android Studio realiza un cambio a la pestaña <your-module> cuando el depurador LLDB encuentra una interrupción en tu código C y C++. También se encuentran disponibles los paneles Frames, Variables y Watches, que funcionan exactamente como lo harían si depuras código Java. Si bien el panel Threads no está disponible en la vista de la sesión de LLDB, puedes acceder a los procesos de tu app por medio de la lista desplegable del panel Frames.

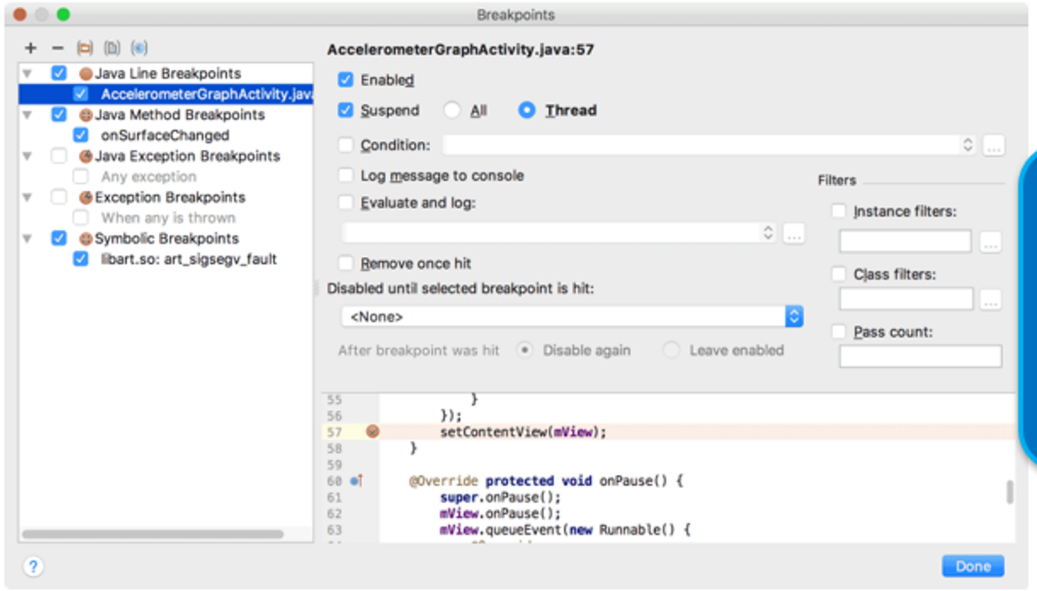
Nota: Mientras inspeccionas una interrupción en tu código nativo, el sistema de Android suspende la máquina virtual que ejecuta el código de bytes Java de tu app, lo que implica que no podrás interactuar con el depurador de Java ni recuperar información sobre el estado de tu sesión de depuración de Java mientras inspeccionas una interrupción en tu código nativo.

2. Android Studio realiza un cambio automático a la pestaña **<your-module>-java** cuando el depurador de Java encuentra una interrupción en tu código Java
3. Durante la depuración con LLDB, puedes usar la terminal LLDB en la vista de la sesión de LLDB para pasar opciones de línea de comandos a LLDB. Si tienes comandos específicos que quisieras que LLDB ejecute cada vez que comiences a depurar tu app (ya sea justo antes o después de que el depurador se adjunte al proceso de tu app), puedes agregarlos a la configuración de depuración.

Durante la depuración de código C o C++, también puedes configurar tipos especiales de interrupciones, llamados puntos de análisis, que pueden suspender el proceso de tu app cuando ésta interactúa con un bloque de memoria específico. Para obtener más información, consulta la sección sobre cómo agregar puntos de análisis.

Cómo ver y configurar interrupciones

Para ver todas las interrupciones y configurarlas, haz clic en View Breakpoints en la parte izquierda de la ventana Debug. Aparecerá la ventana Breakpoints, como se muestra:



En esta ventana "Breakpoints", se enumeran todas las interrupciones actuales y se incluye la configuración de comportamiento para cada una.

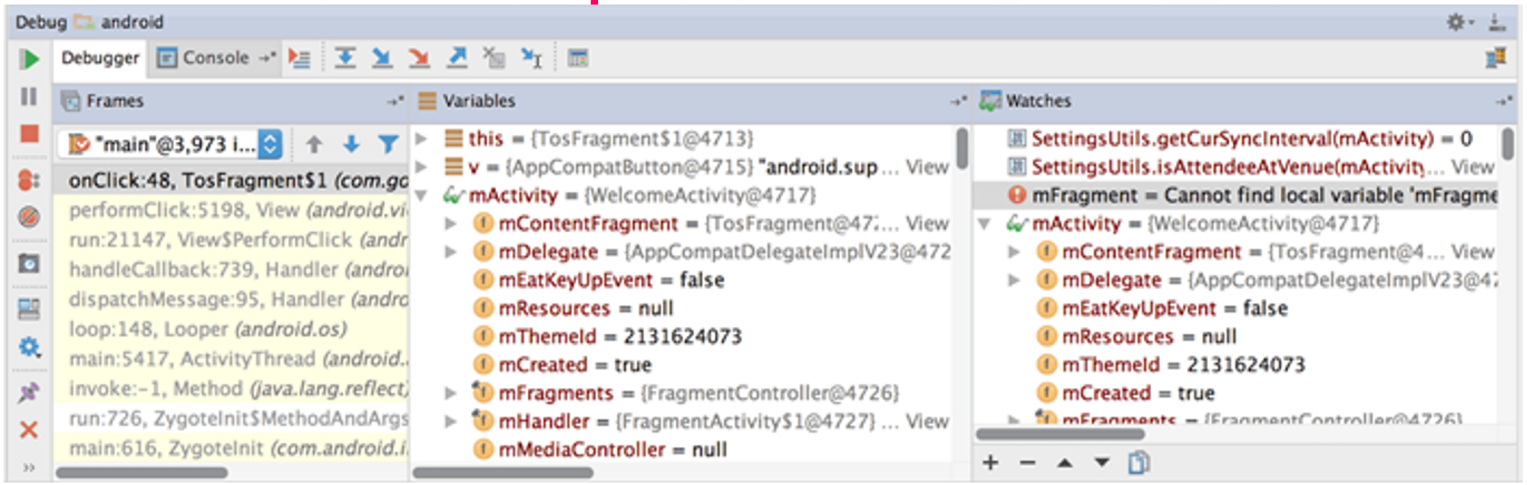
La ventana de Breakpoints

Te permite habilitar o inhabilitar cada interrupción desde la lista de la izquierda.

Si se inhabilita una interrupción, Android Studio no pausa tu app cuando la alcanza. Selecciona una interrupción de la lista para configurarla. Puedes inhabilitar una y hacer que el sistema la habilite luego de que se alcance otra interrupción.

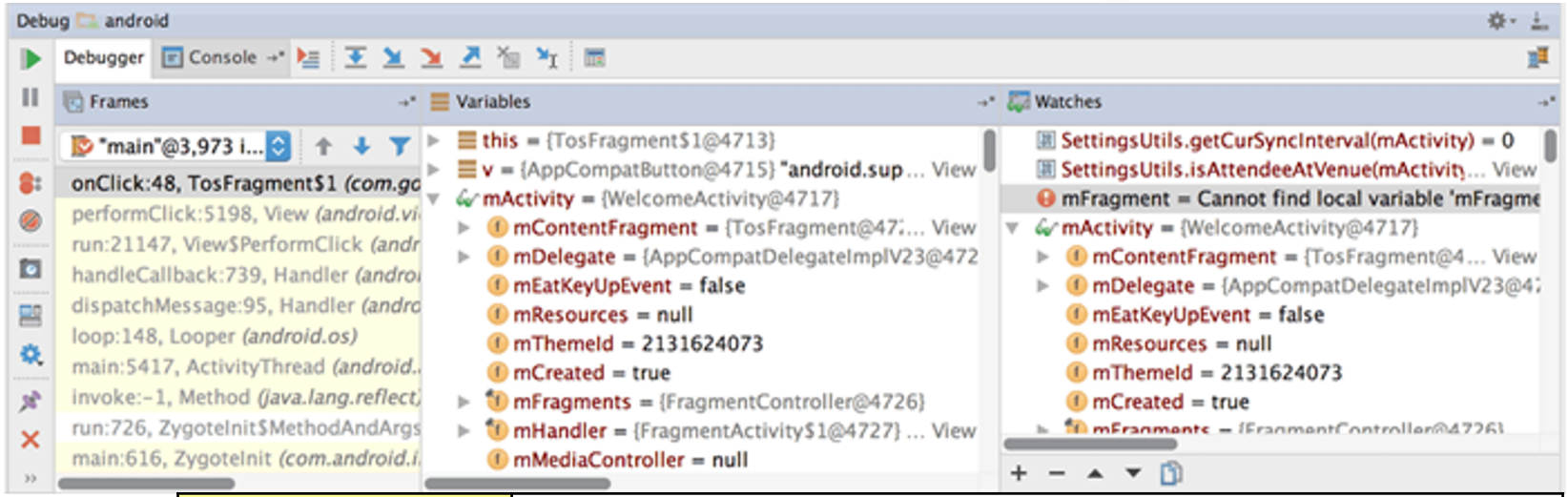
También puedes configurar una interrupción para que se inhabilite una vez que se alcance. Para configurar una interrupción en cualquier excepción, selecciona Exception Breakpoints en la lista de interrupciones.

Cómo inspeccionar variables



Ventana: **Debugger** Panel: **Variables**	Te permite inspeccionar variables cuando el sistema detiene tu app en una interrupción y seleccionas un marco del panel **Frames**. El panel **Variables** también te permite evaluar expresiones ad hoc por medio de métodos estáticos o variables disponibles en el marco seleccionado.

Cómo inspeccionar variables



Ventana: **Debugger** Panel: **Watches**	Se proporciona una funcionalidad similar, con la excepción de que las expresiones agregadas al panel **Watches** se conservan entre sesiones de depuración. Debes agregar controles para variables y campos a los que accedas con frecuencia o que proporcionen un estado útil para la sesión de depuración actual.

Cómo inspeccionar variables

Para agregar una variable o expresión a la lista de **Watches**, sigue estos pasos:

1. Inicia la depuración.
2. En el panel **Watches**, haz clic en **Add** + .
3. En el cuadro de texto que aparece, escribe el nombre de la variable o la expresión que desees controlar y presiona Intro.

Para quitar un elemento de la lista **Watches**, selecciónalo y haz clic en **Remove** − .

Puedes ***volver a ordenar*** los elementos en la lista **Watches**; para ello, selecciona un elemento y haz clic en Up ▲ o Down ▼ .