




- [Inicio](#)
- [¿Quiénes somos?](#)
- [Contacto](#)
- [La AndroTeca](#)
- [Noticias](#)
- [Tutoriales](#)
- [Artículos](#)
- [Entrevistas](#)
- [Recursos](#)

« [ADA Framework Tournament](#)

[WIPJam en el Mobile World Congress 2013](#) »

feb 14 2013



Cómo hacer más cómodo el trabajo con Eclipse

Categorías:

[Tutoriales](#)

por [Rafa Vázquez](#)

Lo parezca o no, trabajar cómodo con nuestro entorno de desarrollo es muy importante de cara a la productividad. Es por eso que nos cuesta tanto cambiarnos a otro IDE cuando estamos acostumbrados a uno concreto. Nos sentimos perdidos, todo lo que es diferente nos parece peor. Porque no hay nada como escribir código cómodamente.



El post de hoy no es tan exclusivo para el desarrollo Android, pero como también nos afecta a nosotros me ha parecido bien tratarlo. Vamos a ver algunos trucos y consejos para escribir código aún más fácilmente, si estamos usando Eclipse.

Todo lo que se comenta en esta entrada son opiniones a título personal y como orientación. Para gustos colores, si no te parecen buenos consejos o tienes otros mejores, cuéntanoslo en los comentarios 😊

Important!

Utilizaré como base el ADT Bundle para Windows que tenemos disponible en la página de desarrolladores, que contiene el SDK y una versión modificada de Eclipse 3.6 con la configuración básica para empezar a desarrollar aplicaciones Android.

Como la lista es larga y variada os dejo un índice para localizar más rápidamente cada cosa:

Preparación del entorno

- [Cambiar la codificación de caracteres](#)
- [Desactivar compilado automático y compilar todo](#)
- [Desactivar corrección ortográfica](#)
- [Explorador de paquetes en vista jerárquica](#)

Ayuda para escribir código

- [Punto y coma siempre al final](#)
- [Auto completado de métodos mejorado](#)
- [Escapar Strings al pegar texto](#)
- [Acciones al guardar](#)
- [Formateo de código](#)
- [Etiquetas](#)
- [Plantillas de código](#)
- [Plantillas de ADT](#)

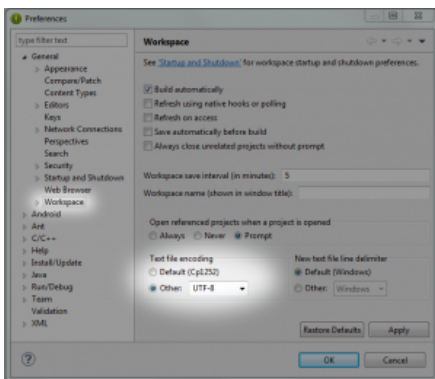
Preparación del entorno

Lo primero que vamos a ver son algunos cambios en la configuración de Eclipse que podemos hacer para facilitarnos el trabajo. Aunque opciones hay muchas, éstas son las que me han parecido más útiles. Mi “*must have*” personal, lo primero que tengo que cambiar si hago una instalación nueva.

Cambiar la codificación de caracteres

Es un tema por el que me he echado las manos a la cabeza más de una vez. Y más de dos, y de tres... Trabajar con diferentes codificaciones es un infierno, sólo causará problemas. Eclipse por defecto en Windows utiliza una codificación, en Linux otra, y en OSX otra distinta. Si trabajas siempre en la misma máquina el problema puede pasar desapercibido, pero en cuanto migras un proyecto de sistema, se lo pasas a alguien, etc. verás como los caracteres especiales se han convertido en símbolos extraños.

Los angloparlantes no suelen tener graves problemas con esto, pues la mayoría de codificaciones *respetan* sus símbolos. Pero nosotros, con tanta tilde y símbolo raro, vamos a encontrarnos en apuros. Lo mejor es curarse en salud y cambiar desde un principio la codificación de todo el workspace, en el menú bajo Window > Preferences, a una universal como UTF-8.

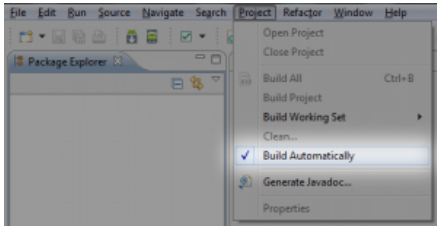


Si lo hacéis no me lo agradeceréis porque no os daréis cuenta de que os ha servido. Pero si no, ya os arrepentiréis. 😊

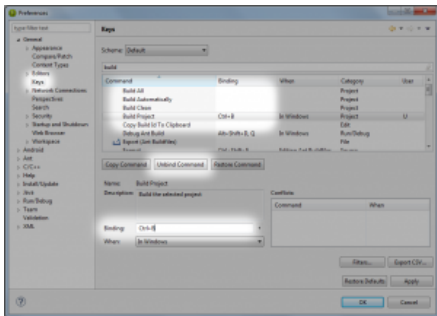
Desactivar *compilado automático y compilar todo*

Por defecto Eclipse tiene activada la opción de compilar automáticamente nuestro proyecto cuando guardamos algún archivo modificado. Esto es especialmente problemático en Android cuando tenemos proyectos relativamente complejos, con varias librerías externas vinculadas y demás compilar puede llevar varios segundos o peor en ordenadores con pocos recursos (¿he oído minutos?). Y no es agradable que nos interrumpan el trabajo sólo por guardar los cambios. Por ello es de lo primero que desactivo cuando abro

Eclipse por primera vez. Sólo hay que desmarcar la opción en Project > Build Automatically.



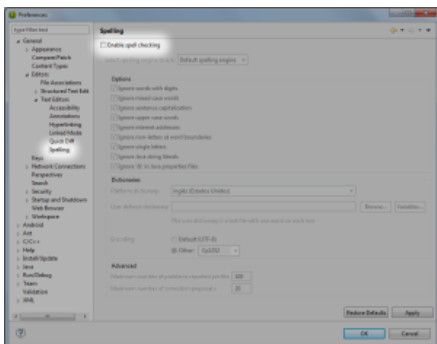
Aunque desactivando dicha opción tenemos el ~~problema~~ inconveniente de tener que compilar manualmente cuando lo necesitemos. Por suerte tenemos el atajo de teclado **Ctrl+B** para hacerlo a golpe de tecla, pero este atajo corresponde al comando *Build All*, es decir que nos compilará todos los proyectos de nuestro workspace. ¡Aún peor que lo anterior! Don't panic, es tan fácil como cambiar el atajo para que ejecute el comando *Build Project* en su lugar, de esa forma la combinación de teclas sólo compilará el proyecto en el que nos encontremos en ese momento. En el apartado Keys de las preferencias buscamos el comando *Build All* y le anulamos el atajo de teclado con **Unbind Command**, luego seleccionamos *Build Project* y en Binding pulsamos la combinación Ctrl+B (u otra) para establecérsela.



Aunque para ejecutar no suele hacer falta compilar porque ya lo hace solo, ya sabemos cómo se pone de tonto a veces Eclipse. Es uno de los comandos que yo más utilizo.

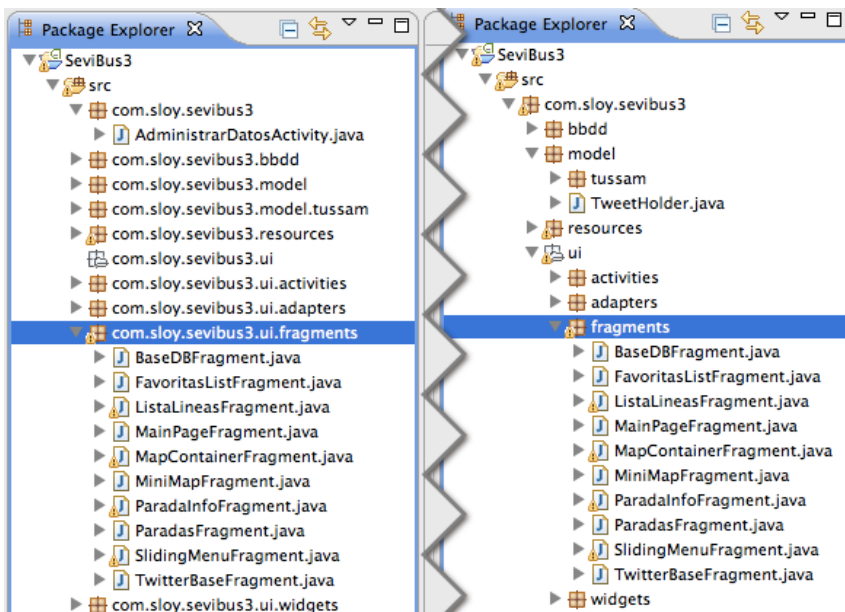
Desactivar corrección ortográfica

Curiosamente Eclipse tiene corrector ortográfico para los comentarios y documentación. Pero a los hispanohablantes no nos sirve de mucho si escribimos en español porque sólo está en inglés. Podríamos descargar e instalar un diccionario en español, pero yo prefiero optar por desactivarlo por completo y quitarme las molestas líneas rojas que salen por todas partes.

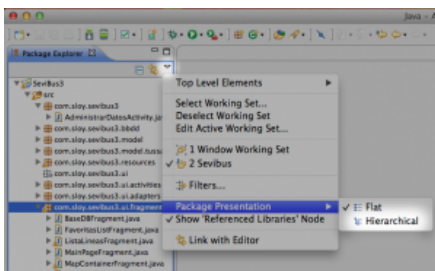


Explorador de paquetes en vista jerárquica

El explorador de paquetes nos permite 2 formas básicas de visualizarlos: **plana** y **jerárquica**. Por defecto nos los muestra de forma plana, pero cuando tenemos muchos paquetes anidados la jerárquica permite verlos más claramente con su estructura de árbol. ¿No os parece?



Podéis usar la que más os guste, en cualquier caso el cambio es sencillo.

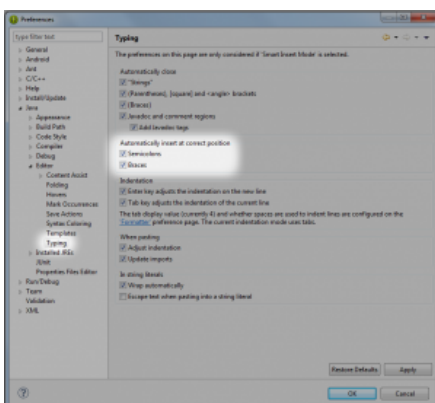


Ayuda para escribir código

Ahora veremos algunos ajustes orientados a facilitar la escritura de código. Eclipse permite un alto nivel de personalización, pero sólo veremos unos que me parecen muy interesantes. Como antes, recomiendo echar un ojo a todas las opciones y adaptarlo todo a nuestras necesidades.

Punto y coma siempre al final


En Java las sentencias llevan un punto y coma al final. Parece una tontería, pero son incontables las veces que he tenido que desplazarme al final de la línea para poner un punto y coma. Con este cambio Eclipse colocará la puntuación correctamente al final de la línea aunque estemos editando en mitad, ahorrándonos un par de pulsaciones de tecla.



Autocompletado de métodos mejorado


El autocompletado de Eclipse es magnífico. **Ctrl+Espacio** es la combinación que más uso con diferencia,

```
nombre.setText(item.getNumero());
```



```
nombre.setText(item.getDescription().Numero());
```

nombre.setText(item.getNumero());



nombre.setText(item.getDescription());

The screenshot shows the 'Preferences' dialog in IntelliJ IDEA, specifically the 'Content Assist' tab. The 'Content Assist' section is expanded, showing various options for code completion and insertion. The 'Show single proposals automatically' checkbox is checked. The 'Show suggested word matches' checkbox is also checked. The 'Show proposed references' checkbox is checked. The 'Show proposed references' checkbox is checked. The 'Show proposed references' checkbox is checked.

Copiar y pegar es el recurso más utilizado. ¿Cuántas veces hemos pegado en una cadena un texto copiado de otra parte que contiene comillas y otros caracteres que necesitan ser escapados (como las comillas en un código html)? Activando esta opción no tendremos que hacerlo más, pues el texto se transformará automáticamente cuando lo peguemos para escapar dichos caracteres.

The screenshot shows the Visual Studio Code Settings application. The left sidebar has 'Text Editor' selected under the 'View' category. The main pane displays the 'Text Editor' settings. The 'Indentation' section is expanded, showing:

- ☒ Use Spaces
- Tab Size: 4
- Indent Size: 4

 The 'Wrapping' section is also expanded, showing:

- Wrap Mode: **Soft**
- ☒ Word Wrap

 At the bottom, there are buttons for 'Restore Defaults', 'Apply', 'OK', and 'Cancel'.

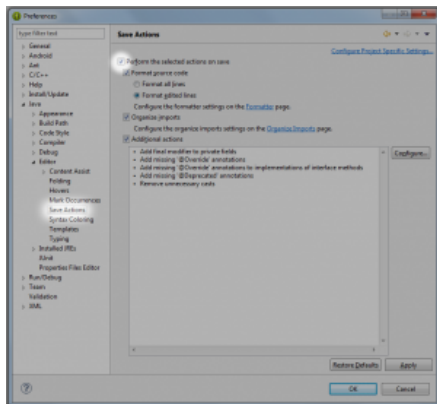
<http://androcode.es/2013/02/como-hacer-mas-comodo-el-trabajo-con-eclipse/>

```
codigoHtml.setText("<a href='\"http://androcode.es/\">AndroCode</a>");

codigoHtml.setText("<a href=\\\"http://androcode.es/\\\">AndroCode</a>");
```

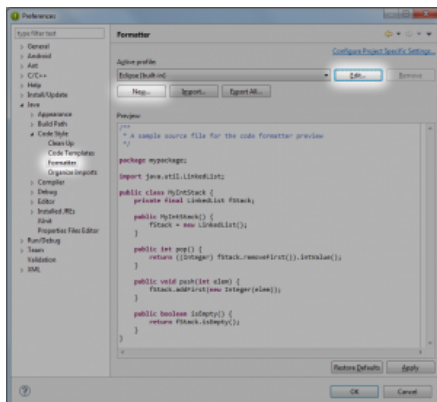
Acciones al guardar

Nos puede venir bien que Eclipse haga determinadas cosas automáticamente cuando guardamos un archivo. Por ejemplo, que reorganice las importaciones de paquetes, o que formatee el código según las reglas que le digamos. Debo confesar que yo personalmente lo tengo desactivado, porque tengo la manía de guardar los cambios constantemente, y al procesar el archivo cada vez puedo notar el retraso en contadas ocasiones si el ordenador está pasando un mal rato y no lo soporto. Pero aun así, si no sois tan maníacos como yo seguro que agradecéis esta función.

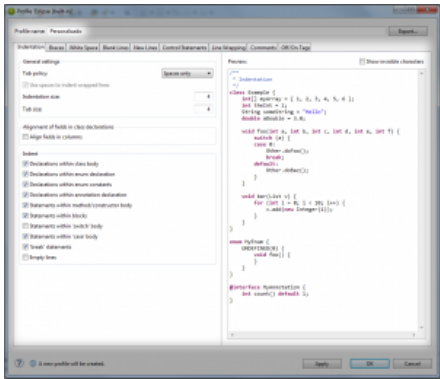


Formateo de código

Otra de las funciones más útiles de Eclipse es el formateo de código automático, mediante la combinación de teclas **Ctrl+Shift+F**. Nosotros escribimos (o pegamos) un churro, con las identaciones mal puestas, espacios sobrantes, etc., y Eclipse nos deja un código limpio, claro y ordenado. Pero es importante para estar cómodos con nuestro IDE que nos produzca código que nos guste. Para ello viene bien echar unos minutos en revisar las opciones de personalización del formateo de código, que no son pocas. Podemos tener varios perfiles, e incluso configurar un perfil distinto para cada proyecto (modificando las preferencias del proyecto concreto, y no las globales). Para personalizarlos creamos uno nuevo o editamos el existente (habrá que guardarlo con otro nombre).



Aquí entran muy en juego los gustos de cada uno. Aconsejo mirar todas las opciones, elegir las que nos parezcan mejores, y volver a mirarlas tras llevar un tiempo con ellas para asegurarnos de que estamos cómodos con todas.

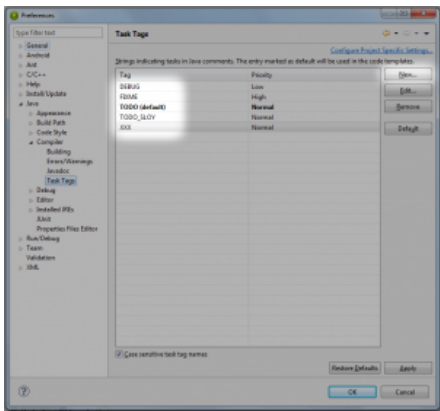


Os comento como ejemplo las que yo suelo cambiar, que son realmente pocas: En *Indentation* pongo Tab Policy a *Tabs Only*, y activo *Statements within switch body*; en *Line Wrapping* pongo el *Maximun line width* a un valor alto como 200 para que no me divida las líneas, y activo *Never join already wrapped lines* por si quiero dividir las manualmente; y por último en *Comments* activo *Never join lines* para decidir yo los saltos de línea, y pongo el *Maximun line width* también a un valor como 200.

Mucho ojo a esta función si trabajamos en grupo con repositorios, o si contribuimos a proyectos de software libre. Debemos adaptarnos a las normas comunes, así que si pensamos dejar que Eclipse formatee nuestro código debemos asegurarnos de respetar dichas normas. No queremos que Git nos marque todo el archivo “en rojo” cuando sólo hemos cambiado una línea, ¿verdad?

Etiquetas

No muy usadas, pero están ahí para ayudar a organizarnos si las necesitamos. Las etiquetas son comentarios especiales que podemos poner para marcar nuestro código. La más famosa es la etiqueta **TODO**, para dejar anotado algo que tenemos que hacer o cambiar en un futuro. Pero nosotros mismos podemos crear nuestras etiquetas personalizadas. Por ejemplo DEBUG para indicar algo que debemos quitar antes de sacar la aplicación a producción, asignar tareas a determinadas personas en un grupo, etc.

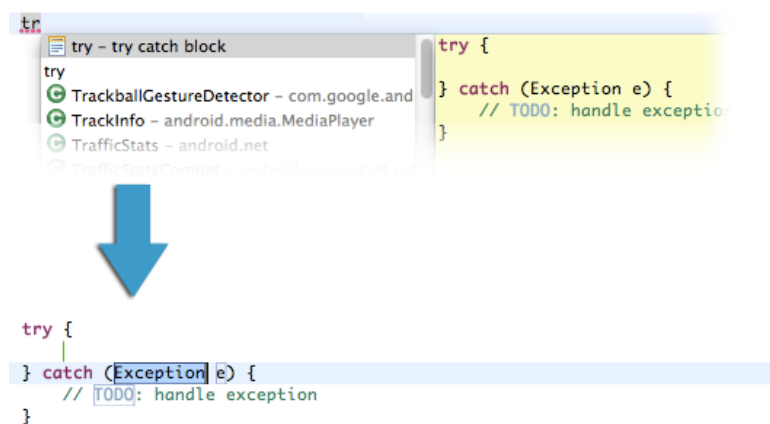


Luego podemos ver la lista de etiquetas usadas en la vista Tasks (Si no la tenemos se activa en Window > Show View > Tasks)

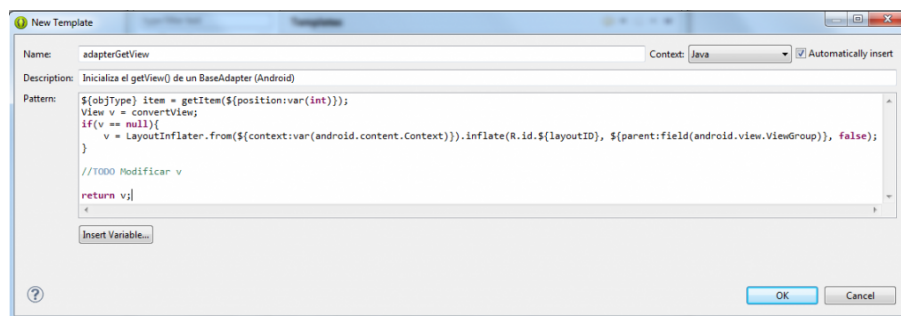
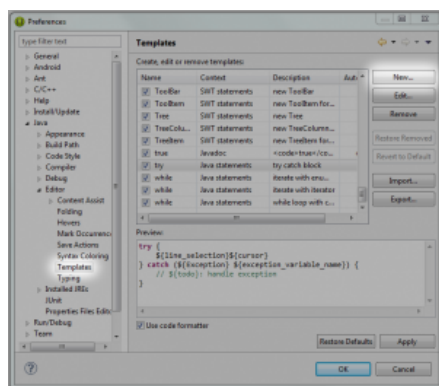
Tasks			
30 items			
	Description	Resource	Path
	TODO ----- hay que hacer algo con lo del currentposition!!!!	HomeActivity...	/SeviBus3/src/com...
	TODO Auto-generated catch block	FavoritasListF...	/SeviBus3/src/com...
	TODO Auto-generated catch block	ListaLineasFr...	/SeviBus3/src/com...
	TODO Auto-generated catch block	ParadaInfoFra...	/SeviBus3/src/com...
	TODO Auto-generated catch block	TwitterBaseFr...	/SeviBus3/src/com...
	TODO Auto-generated method stub	ParadaInfoAct...	/SeviBus3/src/com...
	TODO Crear otras tablas necesarias	DBHelper.java	/SeviBus3/src/com...
	TODO dale caña	DBHelper.java	/SeviBus3/src/com...
	TODO Ejecutar el script con los inserts	DBHelper.java	/SeviBus3/src/com...
	TODO error	TwitterBaseFr...	/SeviBus3/src/com...
	TODO es que de hecho, onStart debería tener más cuidado	FavoritasListF...	/SeviBus3/src/com...
	TODO hacer algo más	ParadasAdapt...	/SeviBus3/src/com...
	TODO la condición debe ser más compleja.	FavoritasListF...	/SeviBus3/src/com...
	TODO localizar stringgggg	ParadasDeLin...	/SeviBus3/src/com...
	TODO los something, dude	DBHelper.java	/SeviBus3/src/com...

Plantillas de código

Eclipse nos permite usar palabras clave para generar automáticamente determinadas estructuras de código, y más aún, crearlas a nuestro gusto. Por ejemplo, uno de los métodos más utilizados en Java es `System.out.println()` para imprimir texto en la consola; pues nos basta con escribir en el editor “sysout” y pulsar el autocompletado (ctrl+espacio) para que nos inserte la línea completa y nos ponga el cursor en la posición del argumento. Lo mismo ocurre con “try” para insertar un bloque try-catch.



La lista completa podemos verla en las preferencias, y ahí mismo podemos añadir nuestras propias plantillas. Por ejemplo, podríamos crear una plantilla así para inicializar el método `getView()` típico del `BaseAdapter`:



Copiar código

Y en el editor de código, poniendo la palabra clave con autocompletado, nos genera el código por el que podemos movernos mediante el tabulador. ¡Magia!


```
public View getView(int position, View convertView, ViewGroup parent) {
    ada
}
//TODO Modificar v
```

adapterGetView - Inicializa el getView() de un
 AdapterViewFlipper - android.widget
 Adapter - android.widget
 AdapterContextMenuInfo - android.widget.Adapter
 AdapterContextMenuInfo - com.actionbarsherlock
 AdapterView - android.widget
 AdapterViewAnimator - android.widget



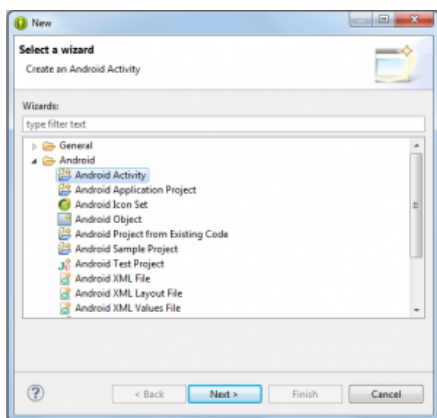
```
public View getView(int position, View convertView, ViewGroup parent) {
    objType item = getItem(position);
    View v = convertView;
    if (v == null) {
        v = LayoutInflater.from(mContext).inflate(R.id.layoutID, parent, false);
    }

    //TODO Modificar v

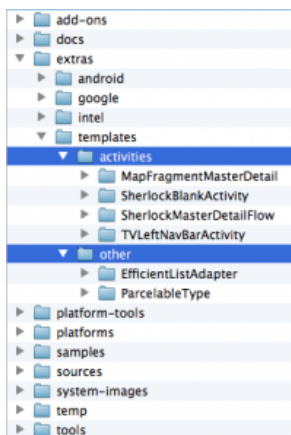
    return v;
}
```

Plantillas de ADT

No hace mucho añadieron al plugin ADT la posibilidad de crear componentes de Android como Actividades para nuestras aplicaciones mediante un **asistente**, que podemos encontrar en el menú *File > New* bajo la categoría de Android, el cual nos genera el código y recursos que podemos editar fácilmente. Por ejemplo, en el caso de las Actividades nos creará un **layout** y **menú** básicos en xml, la **clase Java** correspondiente con el código para mostrar el layout y el menú, y la entrada necesaria en el **AndroidManifest.xml**. E incluso podemos decirle que nos añada elementos extra de navegación como pestañas, ViewPager y más.



Por si fuera poco, un tiempo después añadieron la posibilidad de instalar plantillas personalizadas creadas por nosotros mismos. Yo os recomiendo [estas plantillas](#) creadas por Jeff Gilfelt para crear Actividades compatibles con la magnífica librería [ActionBar Sherlock](#), entre otras. En el repositorio tenéis más información sobre para qué es cada uno de los elementos que añade. Además, instalar las plantillas es tan sencillo como pegarlas en la carpeta **/extras/templates/** del SDK.



Tenéis más información sobre cómo crear plantillas en [este post de Google+](#). ¿Se os ocurren más plantillas útiles? ¿Os animáis a hacerlas y compartirlas? 😊

Y hasta aquí los cambios que os sugerimos para vuestro IDE, espero que algunos os resulten tan útiles como a mí. En futuras entregas intentaremos contar otros trucos a la hora de usar Eclipse. Si tenéis más sugerencias relacionadas son bienvenidas en los comentarios 😊

Fuentes: [Vogella](#) y cosecha propia

¿Te ha gustado?
Compártelo:

Twitter 111

Google

Facebook 73

LinkedIn 14

Google+

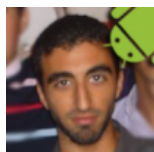


Rafa Vázquez

Seguir

Etiquetas: [ADT](#), [Android](#), [Eclipse](#), [Herramientas](#), [Programación](#), [SDK](#), [Tips](#)

Acerca del autor



Rafa Vázquez

Estudio Ingeniería Informática del Software en Sevilla. Me considero geek sin dinero, amante y desarrollador novato de Android. He creado algunas aplicaciones como SeviBus, TicTacDroide, Kill Bieber y Traductor Hoygan, si es que se puede llamar aplicaciones a estas dos últimas ;) Ganas de aprender más y más no me faltan, e intentaré compartir mis experiencias con vosotros en la medida de lo posible.

22 Comments

Androcode

Login

Recommend 1

Share

Sort by Best



Join the discussion...



josezam • 2 years ago

Hola Sloy, yo quería hacerte una pregunta, cuando inserto código, no me aparece las posibles sugerencias ni haciendo click en ctrl+Space, porque no me aparecen las sugerencias?

Gracias, muy buen trabajo y bastante currado

2 ^ | v • Reply • Share >



Ferran • a year ago

Buenas tardes!

Tengo una duda, la función del punto y coma al final de cada línea, la tengo activada y no funciona. Se supone que inserta un punto y coma cada vez que, por ejemplo, creo una instancia ?

Gracias.

1 ^ | v • Reply • Share >



Victor Montes • 3 months ago

Sos un crack viejo, thanks!!!

^ | v • Reply • Share >



jeral • 5 months ago

Hola que tal , una consulta Saben como hacer que un boton funcione solo uan ves y de ahi desactivarlo gracias..!

^ | v • Reply • Share >



Erick Delzo • 8 months ago

Amigo si aún estás atento aqui, porfa como puedo hacer para agregar código html a un boton en eclipse de android, es para una app en la que quiero ingresar una pagina.

^ | v • Reply • Share >



Guido • a year ago

Excelente!! gracias

^ | v • Reply • Share >

**Caelant** • a year ago

Artículo fantástico

Muchísimas gracias por la ayuda!

Un saludo.

[^](#) | [v](#) • [Reply](#) • [Share](#) >**anonimo183** • a year ago

Gracias.. hace mucho tiempo vi este post, y tube que reinstalar eclipse y no recordaba donde estaba todo para configurarlo... (se me perdió el archivo de las preferencias), ahora nose si conoces cual es el metodo para que con la tecla TAB elija la ayuda de codigo resaltada ?.. normalmente es con ENTER.. pero en visual studio creo que es con TAB y ya me acostumbre.

Gracias

[^](#) | [v](#) • [Reply](#) • [Share](#) >**victor** • a year ago

Buenas, tengo un problema y es que cuando uso el autocompletar con syso me lo configura pra un metodo, y sis escribo System.out.print me da error y no consigo configurarlo para quitar esos errores

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Raul** • a year ago

Geniall !!! me fueron de muchisima ayuda!!

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Monkey D Luffy** • 2 years ago

gracias por los tips, me recordaron que la vida es muy facil y no hay la necesidad de hacer las cosas manualmente... jejeje.. espero ver mas aportaciones.. que posten.. yo conocia el 70% de las k pusieron ...

saludossss

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Jose Antonio Ros** • 2 years ago

Echo en falta en el editor las opciones de Rehacer y Deshacer con memoria de un monton de modificaciones (p.ej. en Dreamweaver), . Para mi es muy util. Es posible trabajarlo aqui?

Muchas gracias.

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Chema** • 2 years ago

Muy util! Sabia que se podían hacer muchas cosas, pero no he conseguido dejarlo a mi gusto nunca.

Gracias!

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Andres** • 2 years ago

Tenía problemas con las tildes, pero con el consejo de cambiar la codificación de caracteres, se solucionó. Mil gracias.

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Sloy** ➔ Andres • 2 years ago

Un placer! Es uno de los problemas que más he tenido desde que uso Eclipse, y el que más me he alegrado de aprender a corregir. Especialmente importante cuando compartes código entre varios sistemas operativos ^

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Fulano Software** • 2 years ago

muy util el autocompletado de texto, me ha pasado muchisimo, gracias! Los demas estan muy bien tambien

[^](#) | [v](#) • [Reply](#) • [Share](#) >**Jiménez Martínez Gabriel** • 2 years ago

Muchas gracias por compartir tus conocimientos.

Saludos.

[^](#) | [v](#) • [Reply](#) • [Share](#) >

**Esteve Camps** • 2 years ago

Hola Sloy, una que es muy útil es el cambio de pestañas. Por defecto está configurado con el Ctrl+Shift+F6 i Ctrl+F6 (creo que son esas combinaciones). A mi me gusta configurarlo con el Ctrl+Tab y Ctrl+Shift+Tab. Hace más ligero el uso.

En cualquier caso, muy buenas propuestas las que haces en este post.

Un saludo ;)

^ | v • Reply • Share >

**Marco Gonzalez Reyes** ➔ Esteve Camps • 2 years ago

Este me parece muy util pero no encuentro el identificador de este comando le agradeceria me dijera como se llama en la lista de comandos de Eclipse

^ | v • Reply • Share >

**Steven Ferreira** ➔ Marco Gonzalez Reyes • 2 years ago

Para los que lo necesiten aun: Preferences -> General -> Keys> y allí se pueden reconfigurar todos los atajos de teclado

^ | v • Reply • Share >

**Sloy** ➔ Esteve Camps • 2 years ago

Cierto, también es bastante práctico ese cambio. Gracias por el apunte ;)

^ | v • Reply • Share >

**Jesús** • 2 years ago

Muy buenos consejos la verdad. Me gusta que Eclipse sea tan configurable y lo de las plantillas ya es brutal. Buen artículo si señor.

^ | v • Reply • Share >

ALSO ON ANDROCODE

WHAT'S THIS?

Contacto

1 comment • 6 months ago

Un 'stack' productivo para el desarrollador android #2, UI

3 comments • 5 months ago

Subscribe

Add Disqus to your site

Privacy

**Artículos más leídos**

Cómo hacer más cómodo el trabajo con Eclipse 1

Monetizando nuestras Apps: AdMob en Android 2

Trabajando con Parcelables 3

Introducción a ActionBarSherlock 4

Primeros pasos con ORMLite 5

AndroCode



Seguir

+1

Tweets

Seguir


AndroCode
 @androcode

18 Mayo

Los Avengers se juntan con Dagger 2, RxJava y Retrofit! Una auténtica batalla por la salvación de Clean androcode.es/2015/05/cuando...

Mostrar resumen


AndroCode
 @androcode

19 mar

Tercera entrega de la serie "Un stack productivo para el desarrollador Android": kcy.me/1reky Esta vez sobre compatibilidad

Mostrar resumen


AndroCode
 @androcode

24 feb

Ya está disponible la segunda parte de

Twitter a @androcode


AndroCode
 906 likes

Like Page

Share












Be the first of your friends to like this



Sitios de interés

- [ActualidadGoogle](#)
- [And.roid.es](#)
- [Andro4all](#)
- [Android Pasion](#)
- [Androideity](#)
- [Curso Android Sgoliver](#)
- [DesAndrOId](#)
- [EAndroid](#)
- [El Androide Libre](#)
- [Genbetadev](#)
- [JavaHispano](#)
- [LiME Creative Labs](#)
- [LinuxHispano](#)
- [Xatakandroid](#)

Redactores

-  [AnderWeb](#) (1)
-  [AntoVillarejo](#) (8)
-  [Breogangf](#) (4)
-  [FedeProEx](#) (9)
-  [Fernando F. Gallego](#) (9)
-  [InmaculadaAlcon](#) (5)
-  [Javier Gutiérrez](#) (1)
-  [JMPergar](#) (65)
-  [Kix2902](#) (10)
-  [Rafa Vázquez](#) (22)
-  [Saúl Molinero Malvido](#) (7)
- [Txus.ballesteros](#) (1)
- [Vierco](#) (1)

Copyright

Excepto allí donde se establezcan otros términos, el contenido de este sitio se publica bajo una [Licencia Creative Commons](#).

Creative Commons Licence BY-NC-ND

- [Volver al inicio](#)

Blog elaborado con [WordPress](#) y el [Tema Graphene](#).

☺