

# Entornos de Desarrollo

02 Entornos de Desarrollo

José Luis González Sánchez



# Contenidos

¿Qué voy a aprender?



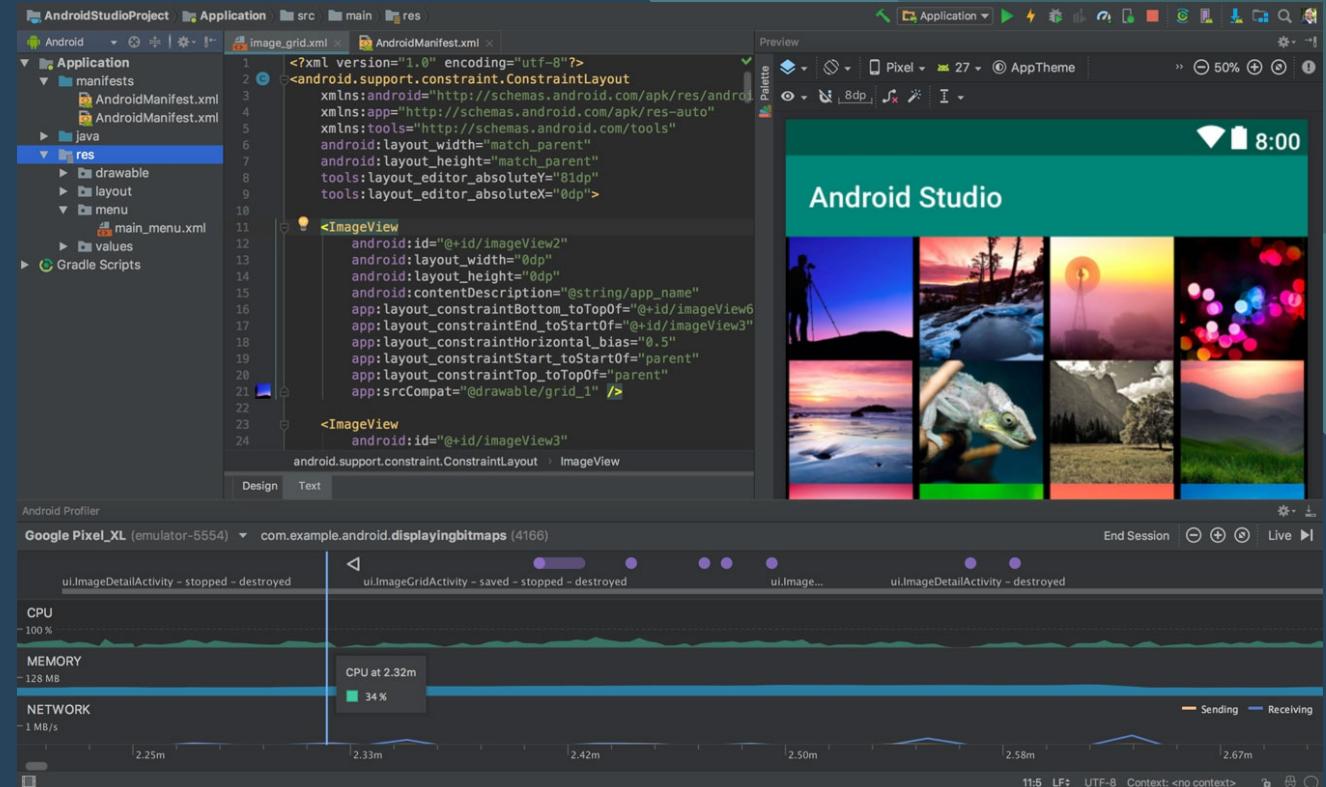
# Contenidos

1. Entornos de desarrollo
2. Funciones y Herramientas



# Entornos de desarrollo

- Un entorno de desarrollo o IDE (Integrated Development Environment) es una aplicación informática que está compuesta por un conjunto de herramientas que facilitan la tarea al programador de forma que ayuda a desarrollar aplicaciones con más rapidez.
- Los entornos de desarrollo son utilizados en la fase de codificación del ciclo de vida de software independientemente del modelo que se utilice.
- Existen entornos de desarrollo para un solo lenguaje o para múltiples lenguajes.





# Funciones y Herramientas

- Un entorno de desarrollo está compuesto por distintas herramientas que nos ayudarán durante todo el proceso de creación del código fuente. Algunas de estas son:
  - **Editor de texto:** Es la herramienta que nos permite escribir el código fuente del programa, ofrece funciones para ayudarnos según estamos desarrollando nuestro código, como son la de resaltar de forma visual palabras reservadas según el lenguaje de programación que estemos utilizando, incluye un analizador léxico que nos corrige en caso de haber escrito mal alguna palabra y un analizador sintáctico que nos informa de si la estructura está bien realizada.

The screenshot shows the Atom code editor interface. On the left is the 'Project' sidebar with a tree view of the project structure, including 'atom', '.git', '.github', 'apm', 'benchmarks', 'docs', 'dot-atom', 'electron', 'exports', 'keymaps', 'menus', 'node\_modules', 'out', 'resources', 'script', 'spec', and 'src'. The main editor area displays the 'text-editor-element.js' file with the following code:

```
Project
src/text-editor-element.js
JS text-editor-element.js
Settings

272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291

getComponent () {
  if (!this.component) {
    this.component = new TextEditorComponent({
      element: this,
      mini: this.hasAttribute('mini'),
      updatedSynchronously: this.updatedSynchronously
    })
    this.updateModelFromAttributes()
  }
  return this.component
}

module.exports =
document.registerElement('atom-text-editor', {
  prototype: TextEditorElement.prototype
})
```

At the bottom of the editor, there are status icons for Babel, master, and a file count indicator (1 file). The status bar also shows the current file path: 'src/text-editor-element.js'.



# Funciones y Herramientas

- **Un compilador:** Es la herramienta encargada de traducir el código fuente que hemos escrito en lenguaje que sea legible para las máquinas (código máquina).

```
<?php
/*
Plugin Name: AT JSON API
Plugin URI:
Description: A Customized API for WordPress
Version: 1.1.1
Author: Achievers Technology
Author URI: http://achieverstechnology.com/
*/
$dir = json_api_dir();
@include_once "$dir/singletons/api.php";
@include_once "$dir/singletons/query.php";
@include_once "$dir/singletons/introspector.php";
@include_once "$dir/models/post.php";
@include_once "$dir/models/comment.php";
@include_once "$dir/models/category.php";
@include_once "$dir/models/tag.php";
@include_once "$dir/models/author.php";
@include_once "$dir/models/attachment.php";
function json_api_init() {
    global $json_api;
    if (phpversion() < 5) {
        add_action('admin_notices', 'json_api_php_version_warning');
    }
}
```

line: 1 / 84 col: 0 sel: 0 INS TAB mode: Unix (LF) encoding: UTF-8 filetype: PHP scope: unknown



# Funciones y Herramientas

- **Un intérprete:** Un intérprete es algo similar al compilador, es decir, su misión es la de traducir en código fuente en código máquina con la diferencia de que el intérprete lo traduce línea a línea según se va ejecutando. Es más lento que el compilador.

The screenshot shows a Java-based IDE interface with two tabs at the top: "RunSelection.scala" and "SandBox.scala". The "SandBox.scala" tab is active, displaying the following Scala code:

```
package test

class SandBox {
  def f() {
    import java.util.ArrayList
    import scala.collection.JavaConversions._
    val array= new ArrayList[Int]
    array.addAll(List(1, 3, 7))
    array.foreach(println(_))
  }
}
```

Below the editor, a toolbar includes icons for Problems, Tasks, Console, and Scala Interpreter (Features). The "Scala Interpreter (Project: Features)" tab is selected, showing the same code in the input field. The output pane displays the results of the execution:

```
import java.util.ArrayList
import scala.collection.JavaConversions...
val array= new ArrayList[Int]
array.addAll(List(1, 3, 7))
array.foreach(println(_))

1
3
7
```

At the bottom, there is an "Evaluate: <type an expression>" input field.



# Funciones y Herramientas

- **Un depurador:** Es una herramienta que nos permite probar y eliminar posibles errores en un programa. Nos permite ejecutar el código línea a línea avanzando o retrasando la ejecución a nuestro gusto, detener el programa en los puntos de ruptura que el desarrollador deseé, ver los datos de variables en el momento de ejecución, etc.

The screenshot shows the Oracle Developer Studio interface during a debugging session. The main window displays the source code of a file named 'quote.cc'. A specific line of code, 'discount = getDiscountFor(customerName);', is highlighted in green, indicating it is currently being executed or has just been executed. The code also includes logic to handle EOF or failure cases and a loop to choose a customer. The bottom pane shows the 'Output - Embedded (Debug)' window displaying the results of the program's execution, which lists various customers and their discounts. The left sidebar contains project navigation tools like 'Projects' and 'Files'.

```
cout << "Enter customer name: ";
getline(cin, customerName);
if (cin.eof() || cin.fail()) {
    exit(EXIT_FAILURE);
}
discount = getDiscountFor(customerName);
if (discount == -1) {
    cout << "Cannot get discount value for customer " << cu
    cout << "Please choose customer from list." << endl;
}
while (discount == -1);
```

Output - Embedded (Debug)

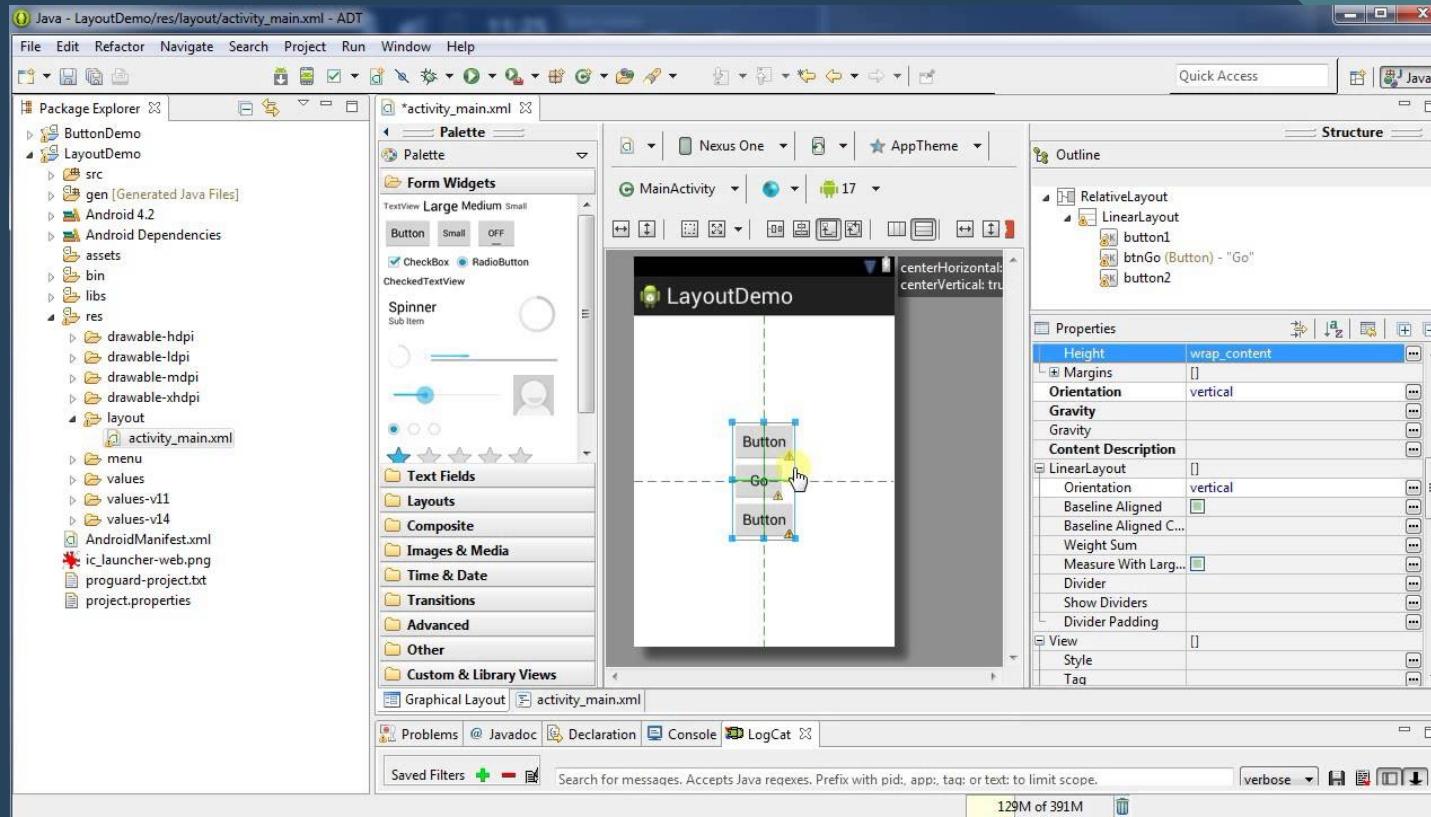
- John has discount 10%
- Mike has discount 0%
- Peter has discount 13%
- Ann has discount 11%
- Tom has discount 9%

Enter customer name: Tom



# Funciones y Herramientas

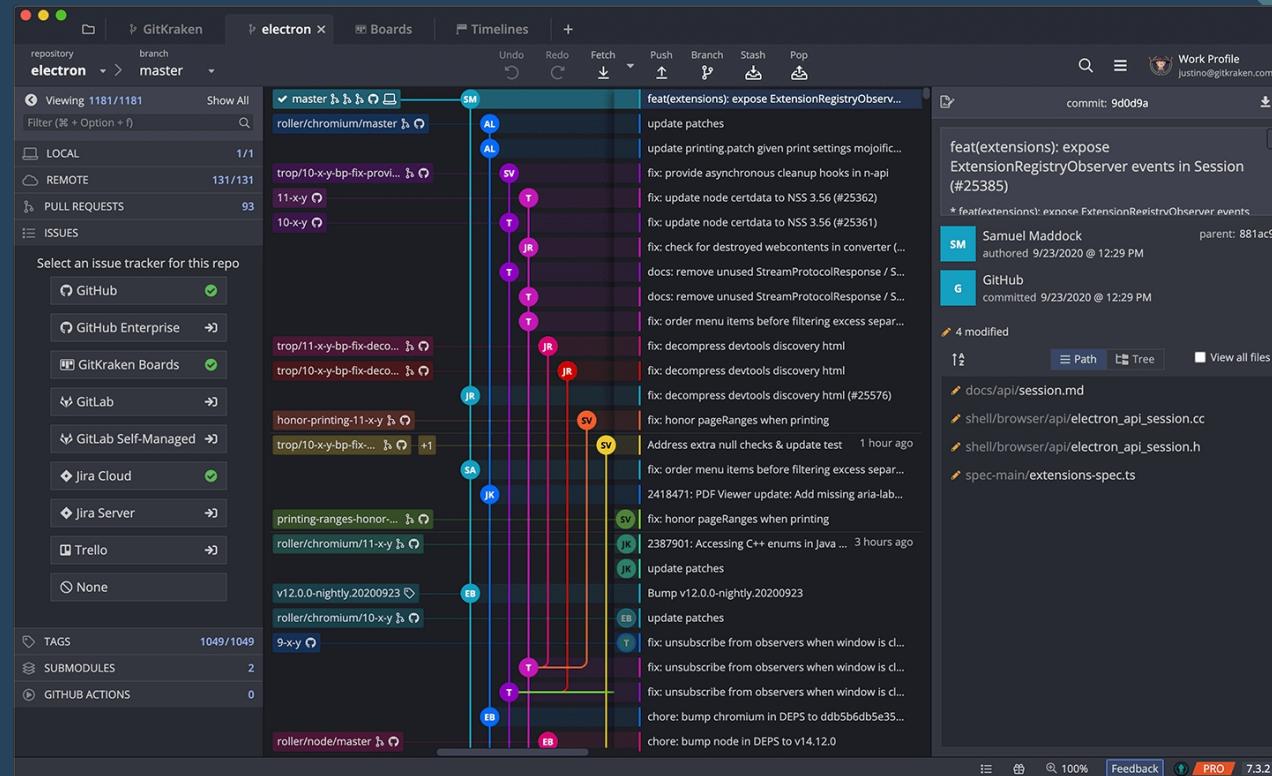
- **Constructor de interfaz gráfica:** Es la herramienta que permite crear y diseñar las interfaces gráficas de forma más intuitiva con las cuales habrá interacción entre la aplicación y el usuario.





# Funciones y Herramientas

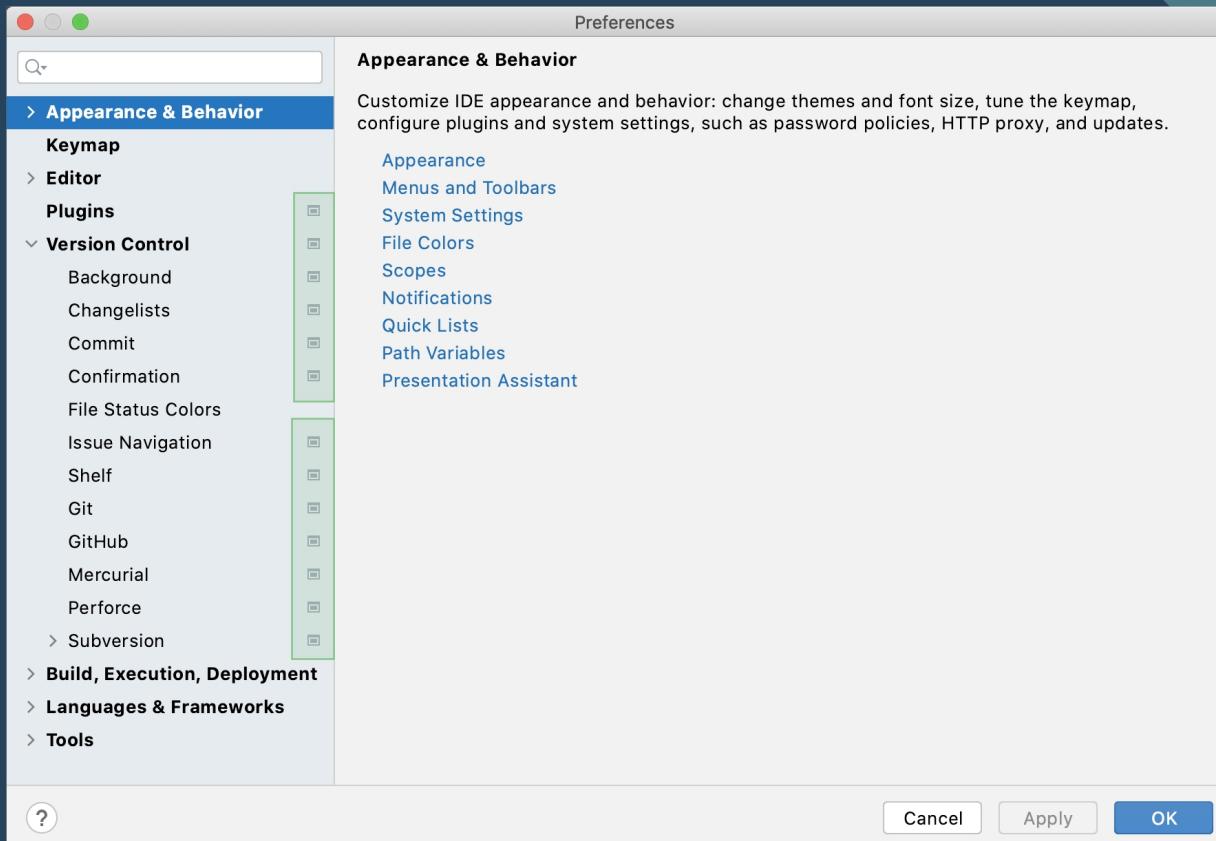
- **Control de versiones:** Es una herramienta que permite al desarrollador controlar los distintos cambios que sufre el código de una aplicación a lo largo de su construcción.





# Funciones y Herramientas

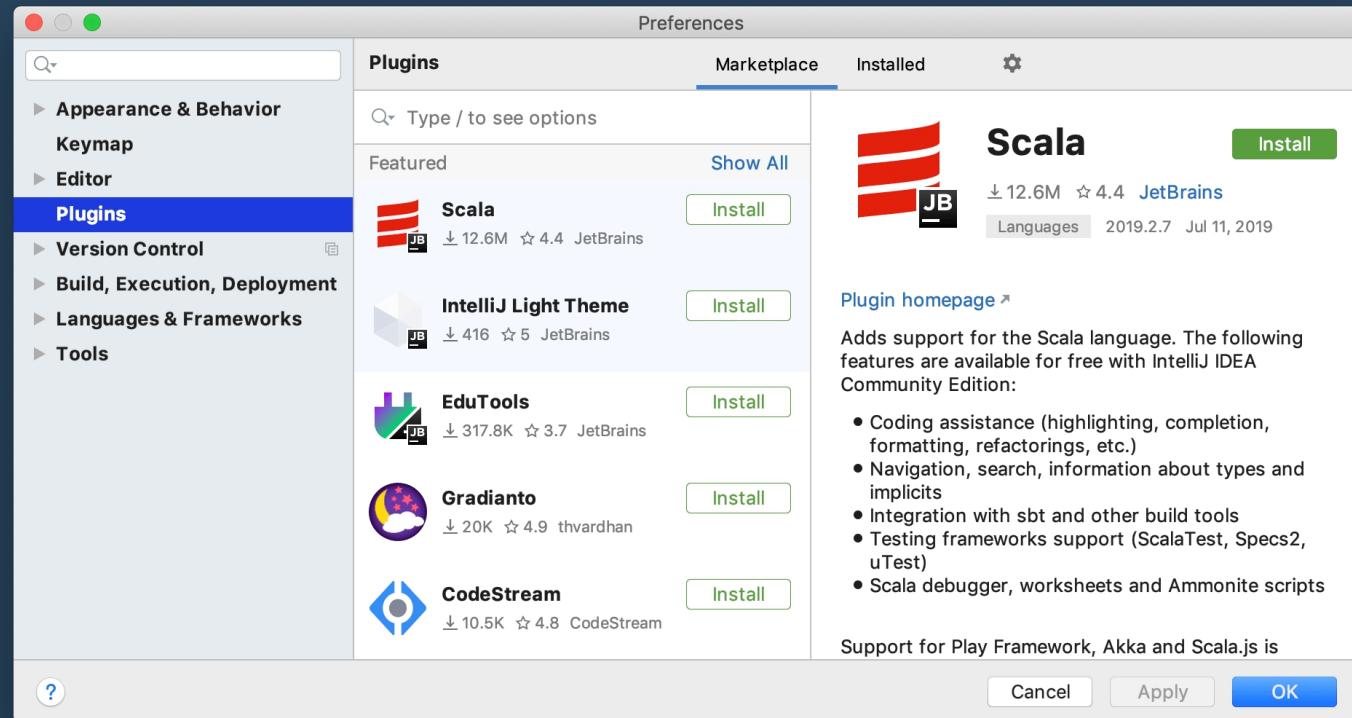
- **Panel de configuración:** Todo entorno de desarrollo dispone de un panel en el que cambiar la configuración del mismo, podemos modificar el aspecto visual, conexiones de red, asignación de teclas, entre muchas otras funciones.





# Funciones y Herramientas

- **Plugins:** son complemento que se relacionan con otras herramientas para agregarle una nueva función y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal.



“

"Los buenos programadores usan sus cerebros, pero unas buenas directrices nos ahorran de tener que hacerlo en cada caso"

- Francis Glassborow

”



# Recursos

- Twitter: <https://twitter.com/joseluisgonsan>
- GitHub: <https://github.com/joseluisgs>
- Web: <https://joseluisgs.github.io>
- Discord: <https://discord.gg/WKKvSJCS>
- Aula Virtual: <https://aulavirtual33.educa.madrid.org/ies.luisvives.leganes/course/view.php?id=246>



# Gracias

José Luis González Sánchez

