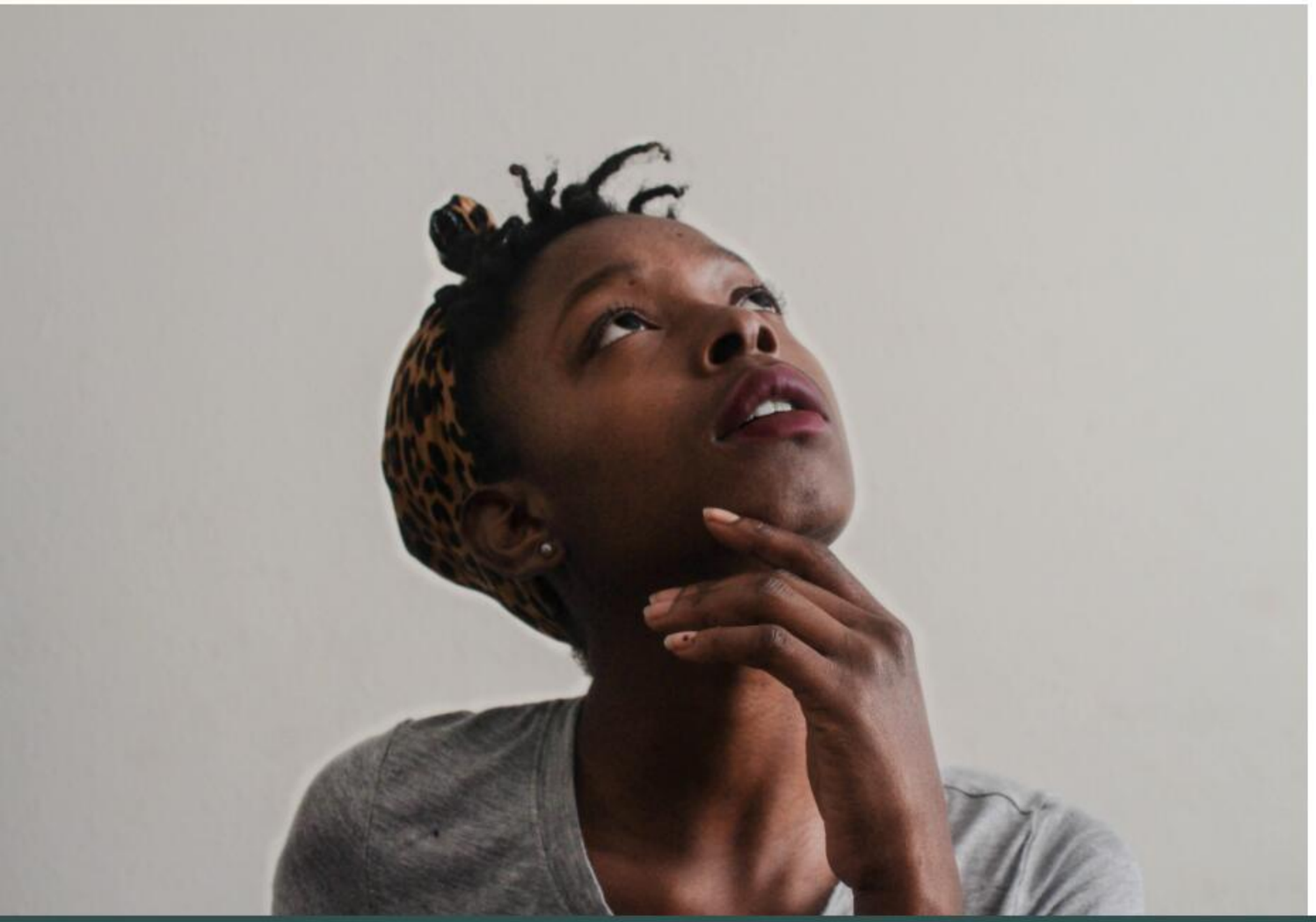
MeLiSA

Herramientas útiles para desarrolladores

Xornada de presentación de software con licencia para uso científico

¿Qué herramientas de software libre puedo utilizar para mi proyecto?

???



Herramientas de software libre

sonarqube

IJ

BACKBONE.JS

apigility by zend framework

bluefish

Vim

Code::Blocks

catalyst Web Framework

jQuery

eclipse

SUBVERSION

ASP.net

Notepad++

CLOUD9

git

NetBeans

DEV C++

appium

wakanda.io

node JS

ZEND FRAMEWORK

Bugzilla

Falcon

LoopBack.io Node.js API Framework

mono

SoapUI

ember

nette FRAMEWORK

CONDITION IZR

GRUNT

prototype

TortoiseSVN

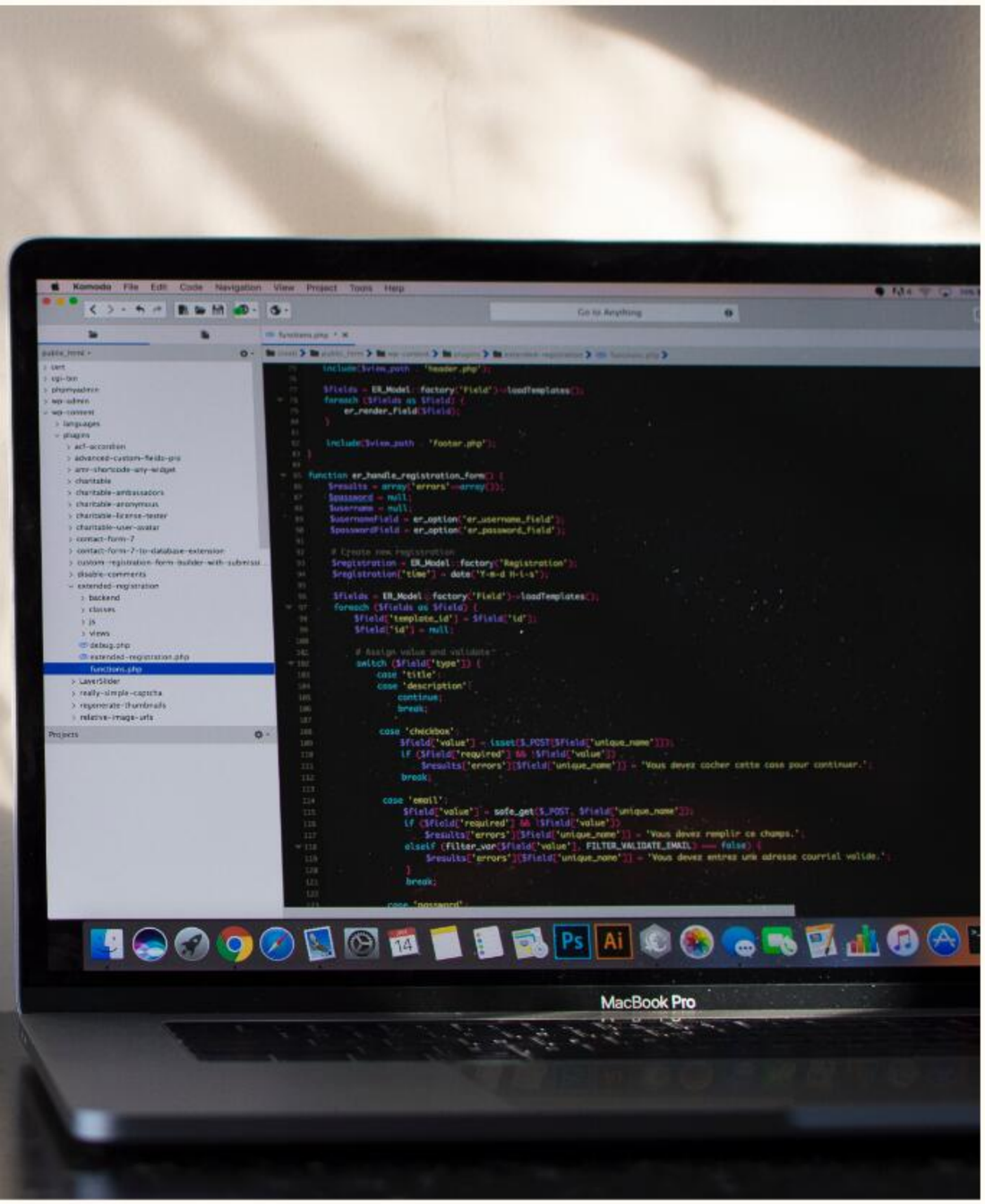
RAILS

Bazaar

Herramientas Útiles

- Entorno de Desarrollo Integrado (IDE)
- Control de Versiones

Entornos de Desarrollo Integrado



¿Qué es un IDE?

Aplicación informática que proporciona servicios integrales para el desarrollo de software

Ventajas

01 Editor de Código Fuente

02 Validación - Error de sintaxis

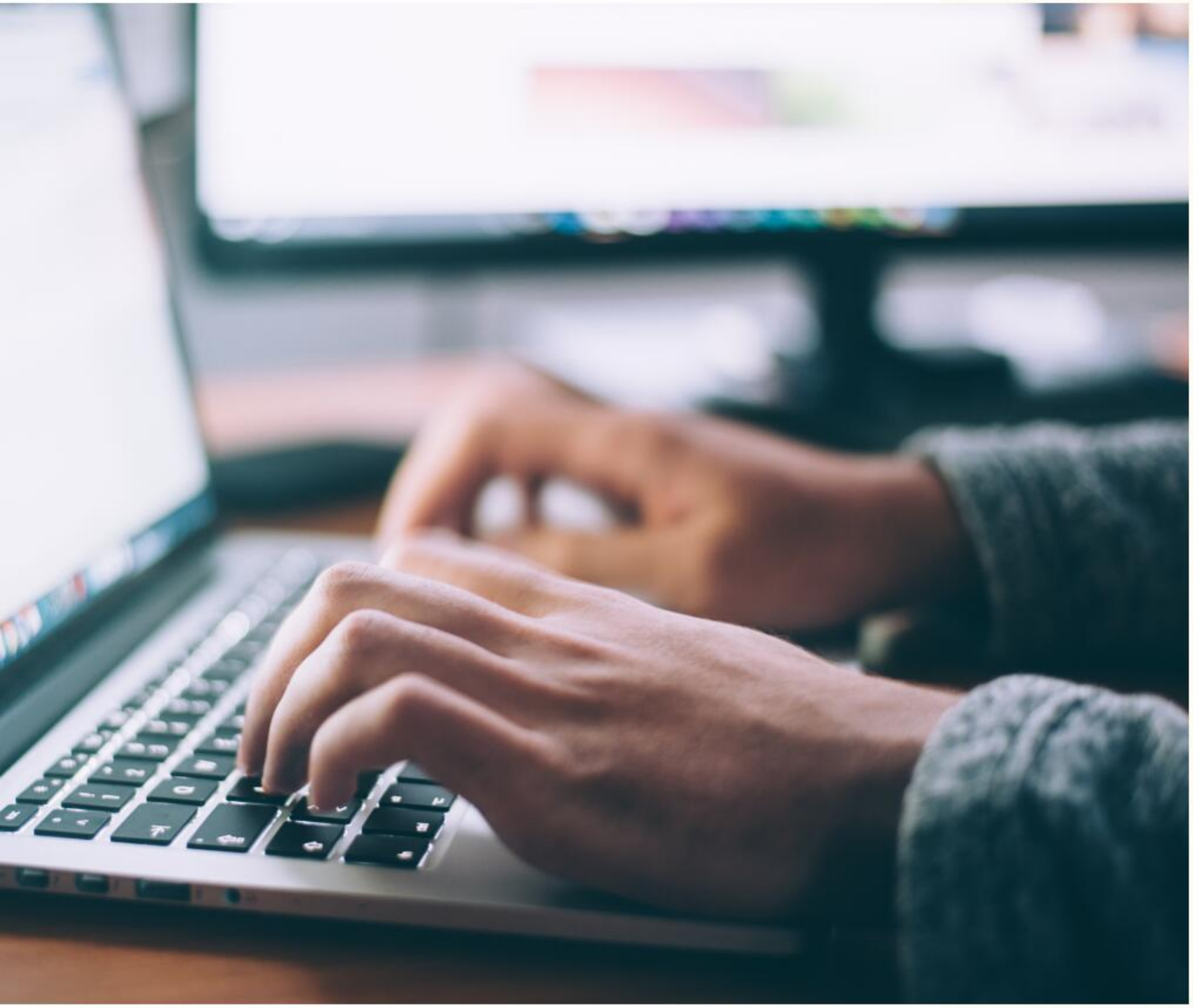
03 Depurador - Degubber

04 Autocompletado - IntelliSense

05 Compilador y/o Interprete

06 Importar y Exportar proyectos

Entornos de Desarrollo Integrado



eclipse

NetBeans

R Studio®

NetBeans

proyectoprueba - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

<default config>

Projects | Files | Services

- proyectoprueba
  - Source Packages
    - proyectoprueba
      - Proyectoprueba.java
  - Test Packages
  - Libraries
  - Test Libraries

main - Navigator

Members <empty>

- Proyectoprueba
  - main(String[] args)

Start Page | Proyectoprueba.java

Source History

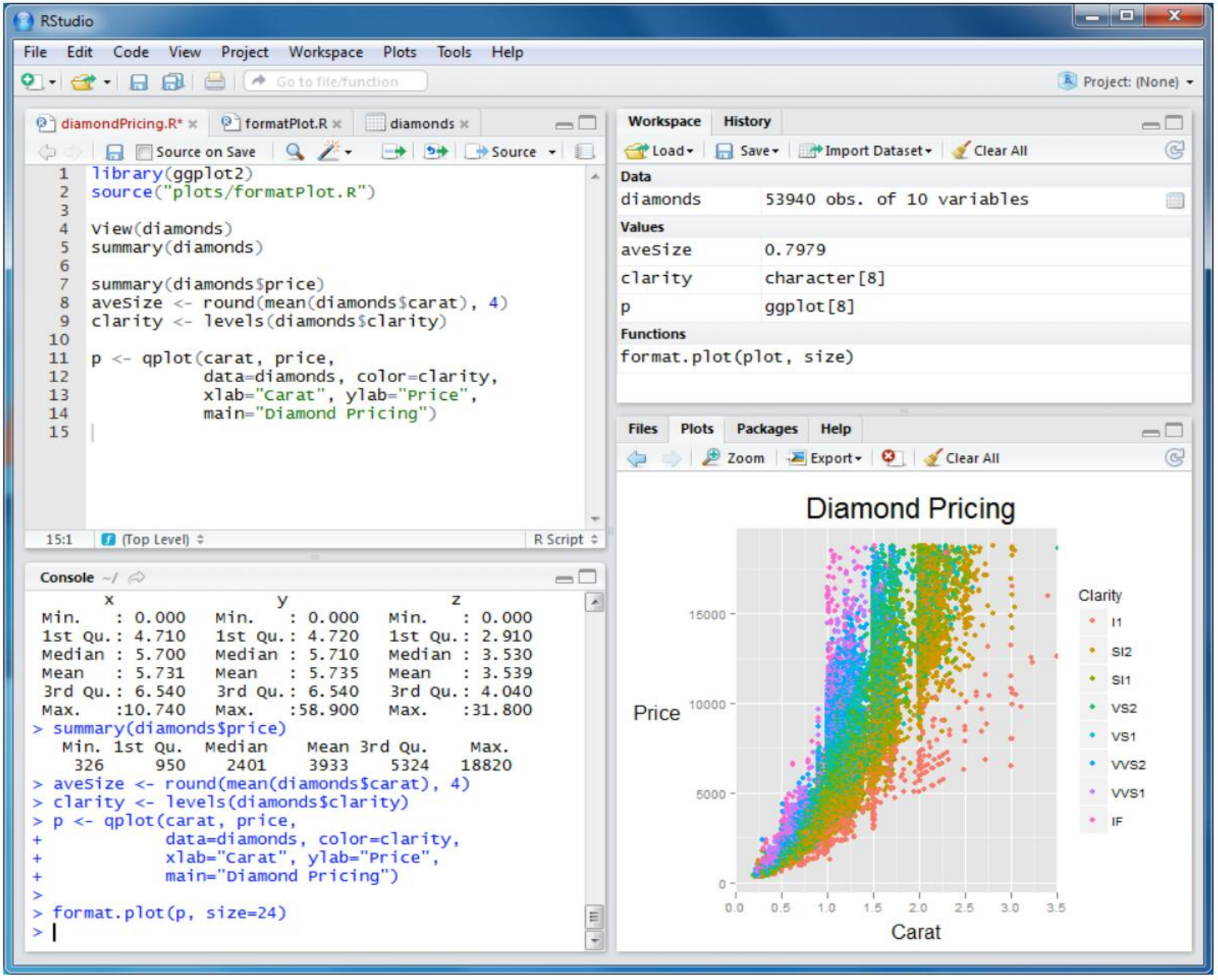
```
12  public class Proyectoprueba {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          // TODO code application logic here
19          String frase = "Hola Mundo";
20          System.out.println(frase);
21      }
22
23  }
24
```

proyectoprueba.Proyectoprueba > main

Output - proyectoprueba (run)

```
run:
Hola Mundo
BUILD SUCCESSFUL (total time: 5 seconds)
```

RStudio


RStudio
File Edit Code View Project Workspace Plots Tools Help
Go to file/function
Project: (None)
diamondPricing.R*
formatPlot.R
diamonds
Source on Save
Source
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 View(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12 data=diamonds, color=clarity,
13 xlab="Carat", ylab="Price",
14 main="Diamond Pricing")
15
15:1 (Top Level) R Script
Console ~/
x y z
Min. : 0.000 Min. : 0.000 Min. : 0.000
1st Qu.: 4.710 1st Qu.: 4.720 1st Qu.: 2.910
Median : 5.700 Median : 5.710 Median : 3.530
Mean : 5.731 Mean : 5.735 Mean : 3.539
3rd Qu.: 6.540 3rd Qu.: 6.540 3rd Qu.: 4.040
Max. :10.740 Max. :58.900 Max. :31.800
> summary(diamonds$price)
Min. 1st Qu. Median Mean 3rd Qu. Max.
326 950 2401 3933 5324 18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+ data=diamonds, color=clarity,
+ xlab="Carat", ylab="Price",
+ main="Diamond Pricing")
>
> format.plot(p, size=24)
>
Workspace History
Load Save Import Dataset Clear All
Data
diamonds 53940 obs. of 10 variables
Values
aveSize 0.7979
clarity character[8]
p ggplot[8]
Functions
format.plot(plot, size)
Files Plots Packages Help
Zoom Export Clear All
Diamond Pricing
Price
Carat
Clarity
I1
SI2
SI1
VS2
VS1
VVS2
VVS1
IF
0
5000
10000
15000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

Control de Versiones - VCS

Aplicación para gestionar cambios en el códifo fuente y poder revertirlos

Ventajas

01 Registra los cambios sobre uno o varios archivos

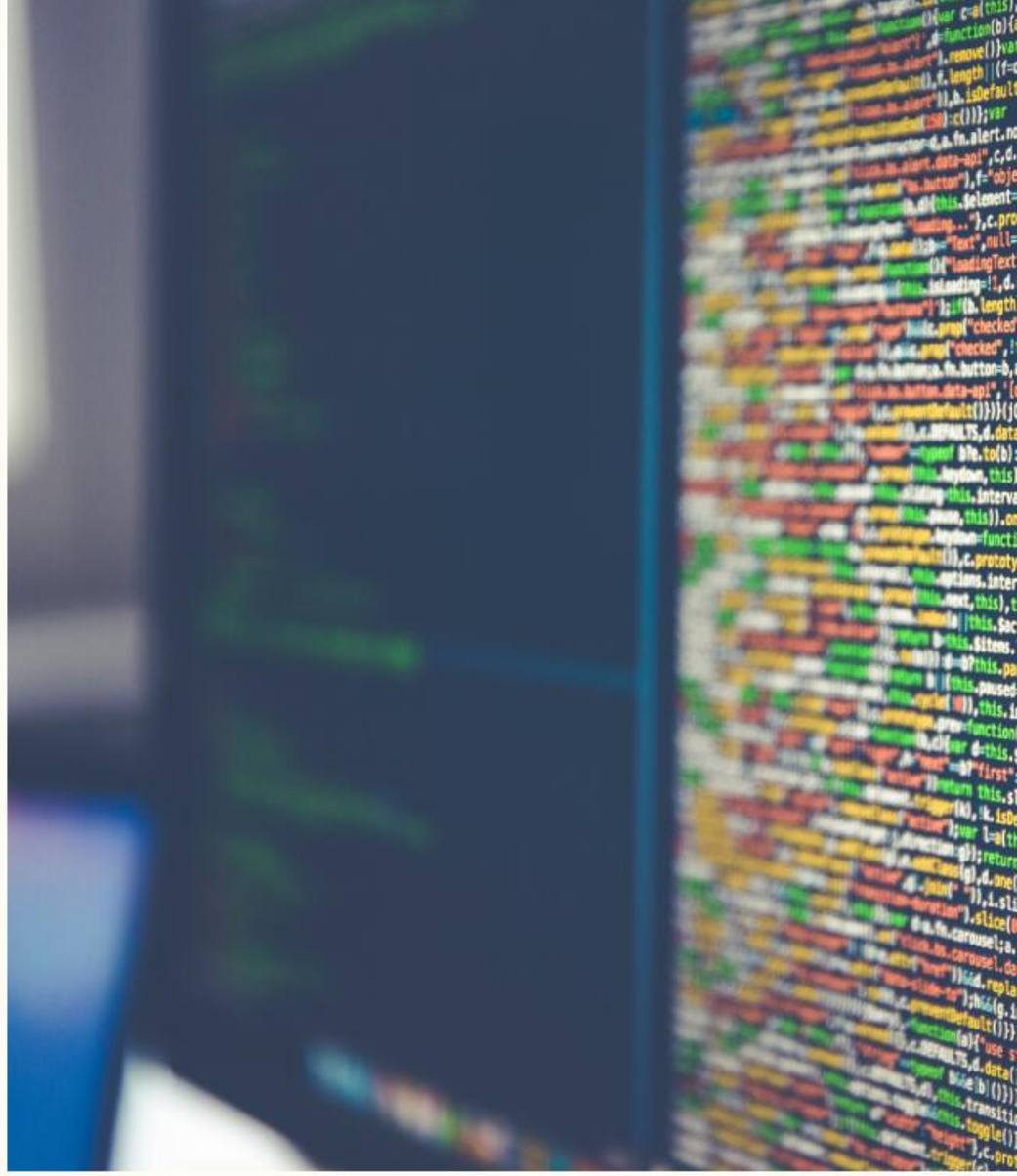
02 Permite recuperar las versiones anteriores

03 Permite revertir el proyecto entero

04 Comparar cambios

05 Identificar quién realizó los cambios

06 Recuperar el archivo en caso de perdida



Control de Versiones - VCS

Uno de los SCV más populares

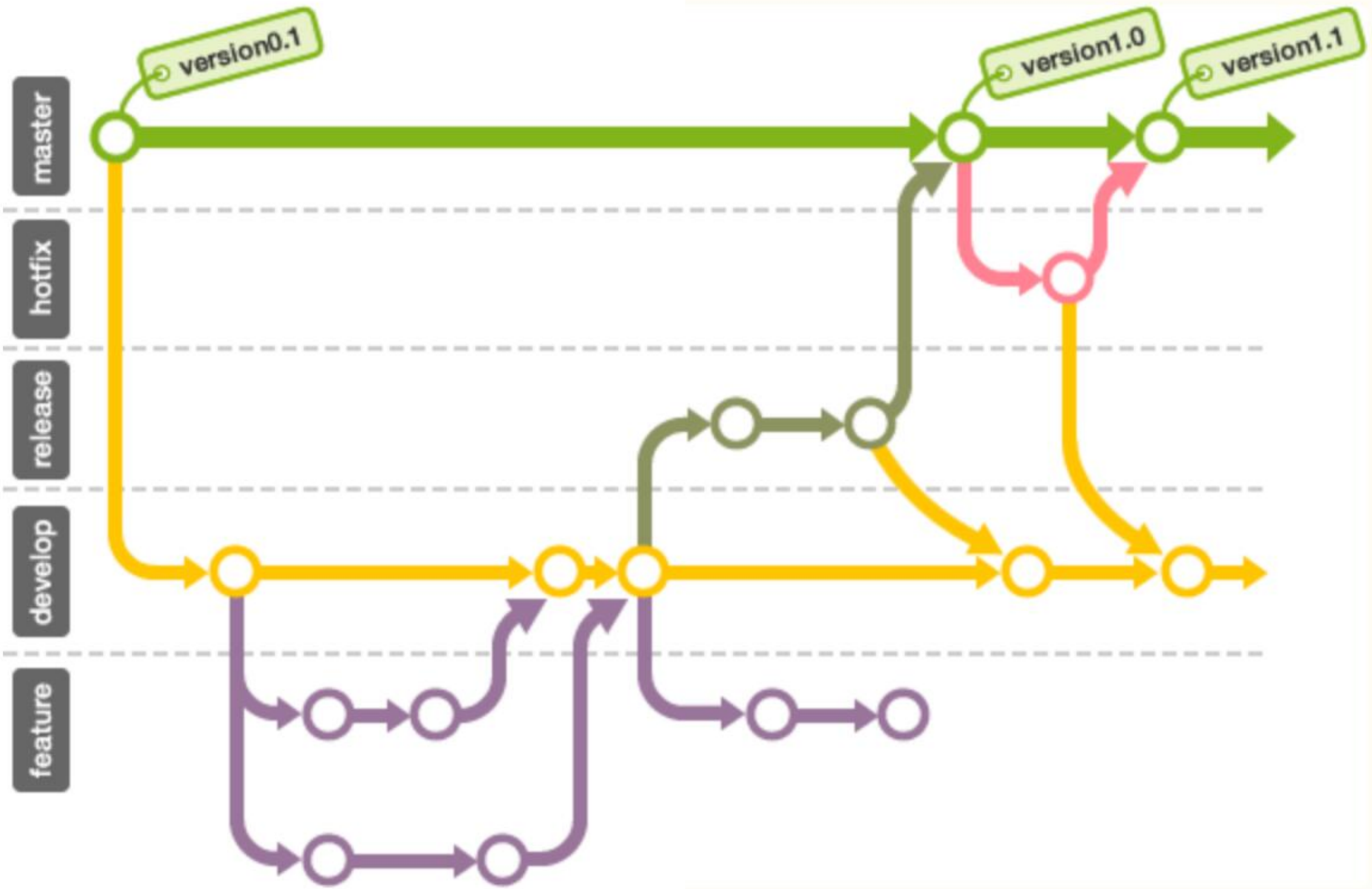
SCV Distribuido:

- Utiliza ramas favoreciendo el trabajo en paralelo sobre el msmo proyecto
- Permite a varias personas trabajar a la vez
- Se genera una rama maestra de donde se extienden otras ramas
- Cuando hay cambios en una rama permite que se combine con otras ramas fusionándose en la rama maestra (merge)

Comandos Básicos

01	git init	→	Crea un nuevo repositorio git
02	git add.	→	Comprueba si hay cambios
03	git status	→	Comprueba el estado del repositorio actual
04	git commit	→	Subida de archivos al repositorio local No se propaga el cambio
05	git pull	→	Actualiza la versión del código (update)
06	git push	→	Envía los cambios locales (commit) al resto de nodos
07	git log	→	Ver historial de los cambios
08	git diff	→	Cambios realizados en un archivo

git


version0.1
version1.0
version1.1
master
hotfix
release
develop
feature

GitLab

¿Qué es GitLab?

Servicio web de control de versiones y desarrollo de software colaborativo basado en Git

GitLab vs GitHub

Similitudes

- Servicio web para Git
- Tienen interfaz web
- Proveen repositorios privados
- Plataformas para compartir

Diferencias con GitHub

- Tiene licencia Open Source
- Permite repositorios privados ilimitados
- Permite instalarlo en nuestro propio servidor
- Tiene Auto DevOps
- Tiene seguimiento de incidencias

GitLab

¿Quiénes usan GitLab?

IBM

SONY

Alibaba.com™

CERN

NASA

¡ GRACIAS !

Juan Arroyo

Xornada de presentación de software con licencia para uso científico