E.S. Ingeniería y Tecnología

Ingeniería Informática

Lenguajes y Sistemas Informáticos

Lenguajes y Paradigmas de Programación
Práctica de laboratorio **#3**

1. Iniciar una sesión de trabajo en GNU-Linux.

2. Abra una terminal.

3. Muestre el árbol de directorios de su HOME.(`tree`)

4. Sitúese en el **directorio** de la asignatura "Lenguajes y Paradigmas de Programación" esto es en el directorio *LPP*. (`cd LPP`)

5. Muestre el contenido del directorio actual. (`ls -la`)

6. Cree un nuevo directorio denominado *rebase* (`mkdir rebase`). Este será el **directorio de trabajo** durante la realización de esta práctica.

7. Sitúese en el directorio *rebase*. (`cd rebase`)

8. Ponga el directorio *rebase* bajo el control de versiones, es decir, cree un repositorio *git*. (`git init`)

9. Con un editor de textos cree el fichero `.gitignore` en el directorio de trabajo. Este fichero ha de contener la expresión regular que evita almacenar los ficheros acabados en el símbolo ~ denominado suprasegmento, o tilde de la letra ñ.

10. Muestre el estado del repositorio `git` local. (`git status`)

11. Añada todos los ficheros del directorio actual al *índice de git*. (`git add .`)

12. Confirme (*commit*) los cambios. (`git commit -m "Creando el fichero .gitignore"`)

13. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate`)

14. Con un editor de textos cree el fichero `README.md` en el directorio de trabajo. Este fichero ha de contener: El nombre del alumno y el número y título de la práctica.

15. Añada todos los ficheros del directorio actual al *índice de git*. (`git add .`)

16. Confirme (*commit*) los cambios. (`git commit -m "Creando el fichero README.md"`)

17. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --oneline --decorate --date=relative --all`)

18. Cree un fichero `respuestas.txt` en el que responda las preguntas de esta práctica de laboratorio.
    (`echo "Respuestas de Primeros Pasos en Ruby" > respuestas.txt`)

19. Añada todos los ficheros y subdirectorios del directorio actual al *índice de git*. (`git add .`)

20. Confirme los cambios del índice. (`git commit -m "Creado el fichero de respuestas"`)

21. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

22. Acepte la tarea asignada en `GitHub Classroom` denominada "Primeros pasos con Ruby".

23. Copie la dirección `SSH` del repositorio remoto en *GitHub* de la práctica .
    (`git@github.com:ULL-ESIT-LPP-1920/rebase-aluXXX.git`)

24. En la consola, cree un enlace al repositorio remoto con nombre corto *ghp03*
    (`git remote add ghp03 git@github.com:ULL-ESIT-LPP-1920/rebase-aluXXX.git`)

25. Empuje los cambios en el repositorio remoto denominado *ghp03*. (`git push -u ghp03 master`)

26. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

27. Cree una rama *doc*. (`git branch doc`)

28. Muestre la rama activa. (`git branch`)

29. Sitúese en la rama *doc*. (`git checkout doc`)

30. Muestre la rama activa. (`git branch`)

31. Dibuje el grafo de las confirmaciones hasta el momento. ¿Cuántas ramas hay?¿Cuáles son locales y cuáles son remotas? Escriba la respuesta en el fichero creado en el ejercicio 18.

32. Añada todos los ficheros y subdirectorios del directorio actual al *índice de git*. (`git add .`)

33. Confirme los cambios del índice. (`git commit -m "Grafo con la rama doc"`)

34. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

35. Sitúese en la rama *master*. (`git checkout master`)

36. Muestre la rama activa. (`git branch`)

37. Con un editor de textos escriba el código fuente Ruby que hace que se muestre por la consola la frase `"Hola Mundo"`, almacénelo en el fichero `helloworld.rb`

38. Añada todos los ficheros y subdirectorios del directorio actual al *índice de git*. (`git add .`)

39. Confirme los cambios del índice. (`git commit -m "Creando Hola mundo en Ruby"`)

40. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

41. Sitúese en la rama *doc*. (`git checkout doc`)

42. Dibuje el grafo de las confirmaciones hasta el momento. Indique dónde se encuentra el HEAD. Escriba la respuesta en el fichero creado en el ejercicio 18.

43. Añada todos los ficheros y subdirectorios del directorio actual al *índice de git*. (`git add .`)

44. Confirme los cambios del índice. (`git commit -m "Grafo con Hello World en Ruby añadido"`)

45. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

46. Reorganice (rebase) los cambios de la rama *doc* en la rama *master*. (`git rebase master`)

47. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

48. Mediante un avance rápido lleve la rama *master* hasta la rama *doc*.
    (`git checkout master; git merge doc`)

49. Dibuje el grafo de las confirmaciones hasta el momento. Indique dónde se encuentran las ramas *doc* y *master*. Escriba la respuesta en el fichero creado en el ejercicio 18.

50. Añada todos los ficheros y subdirectorios del directorio actual al *índice de git*. (`git add .`)

51. Confirme los cambios. (`git commit -m "Grafo con el resultado de la reorganización"`)

52. Compruebe que el intérprete `Ruby` está disponible (`ruby --version`)

53. Si es necesario instalar o actualizar el intérprete de `Ruby` utilice `rvm`.

a) Compruebe que la herramienta `RVM` (*Ruby Version Manager* - `https://rvm.io/`) está instalada (`rvm --version`)

b) Si es necesario instalar `rvm` ejecute (

```
\curl -sSL https://get.rvm.io | bash -s stable
```

)

c) Si se produce el error "`Imposible comprobar la firma: clave pública no encontrada`" ejecute (

```
gpg --keyserver hkp://pool.sks-keyservers.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3
```

)

y si falla

(

```
command curl -sSL https://rvm.io/mpapis.asc | gpg --import -
command curl -sSL https://rvm.io/pkuczynski.asc | gpg --import -
```

)

y de nuevo (

```
\curl -sSL https://get.rvm.io | bash -s stable
```

)

d) Configure `rvm` ejecutando (

```
‘source /home/usuario/.rvm/scripts/rvm‘
```

)

54. Muestre la información disponible del sistema gestor de versiones de Ruby (RVM - *Ruby Version Manager*). (`rvm info`)

55. Muestre la versión del sistema gestor de versiones de Ruby - RVM. (`rvm -v`)

56. Actualice a la versión estable de RVM más reciente. (`rvm get stable`)

57. Muestre la lista de intérpretes Ruby disponibles para su instalación. (`rvm list known`)

58. Muestre la lista de intérpretes Ruby que ya están instalados. (`rvm list`)

59. Si la lista de intérpretes Ruby instalados es vacía, instale las versiones superiores a la 2.6.

    (

    ```
    rvm install ruby-2.6.3
    rvm install ruby-2.7
    ```

)

60. Compruebe la lista de intérpretes Ruby que ya están instalados. (`rvm list`)

61. Establezca que quiere utilizar la versión 2.6.3 del intérprete Ruby. (`rvm use ruby-2.6.3`)

62. Muestre la versión del intérprete de Ruby disponible. (`ruby -v`)

63. Vuelva a la versión por defecto del intérprete Ruby. (`rvm default`)

64. Muestre la versión del intérprete de Ruby disponible. (`ruby -v`)

65. ¿Qué es RVM? ¿Qué versión se ha instalado? Escriba la respuesta en el fichero creado en el ejercicio 18.

66. Añada todo el contenido del directorio al *índice del repositorio git*. (`git add .`)

67. Confirme los cambios. (`git commit -m "Usando RVM - Ruby Version Manager"`)

68. A diferencia de los lenguajes compilados, existen dos formas de ejecutar Ruby. Se pueden crear ficheros y ejecutarlos con el intérprete desde la línea de comandos, y también, se puede introducir el código de forma interactiva, utilizando un REPL (*Read Eval Print Language*). Ejecute el intérprete interactivo de Ruby (*Interactive Ruby*) (`irb`)

69. ¿Cuál es el resultado de la siguiente operación?

```
2.6.3 :001 > puts "Hello world"
 ???
=> ???
2.6.3 :002 >
```

70. Salga de la sesión interactiva. Para ello escriba el caracter de final del fichero del sistema operativo. (`Ctrl + D`)

71. Ejecute el fichero `helloworld.rb` con el intérprete interactivo. (`irb helloworld.rb`)

72. ¿Cuáles son las diferencias entre la ejecución del programa `"Hola Mundo"` con el intérprete de Ruby (`ruby helloWorld.rb`) y el intérprete interactivo del ejercicio anterior? Escriba la respuesta en el fichero creado en el ejercicio 18.

73. Añada todo el contenido del directorio al *índice del repositorio git*. (`git add .`)

74. Confirme los cambios. (`git commit -m "Comparando las ejecuciones interactiva y normal"`)

75. Con un editor de textos cree un fichero `byebyeWorld.rb` que contenga el siguiente código Ruby. Nótese que en la línea número dos se indica que se quiere utilizar una biblioteca, en concreto la gema (*gem*) PRY.

```
 1  #byebyeWorld.rb
 2  require 'pry'
 3
 4  #define a method
 5  def byebye() puts "bye bye world!!!" end
 6
 7  #set x to 10
 8  x = 10
 9
10  #start a REPL session
11  binding.pry
12
13  #program resumes here (after pry session)
14  puts "program resumes here. Value of x is: #{x}."
```

76. Installe la gema PRY. (`gem install pry`)

77. Ejecute el programa `byebyeWorld.rb` para que se lance PRY. (`ruby byebyeWorld.rb`)

78. ¿Cuál es el resultado de cada una de las siguientes operaciones?

```
[1] pry(main)> puts x
???
=> ???
[2] pry(main)> def hello
[2] pry(main)*   puts "Hello world"
[2] pry(main)* end
=> ???
[3] pry(main)> hello
???
=> ???
[4] pry(main)> byebye
???
=> ???
[5] pry(main)> x = "changed"
=> ???
[6] pry(main)> exit
program resumes here. Value of x is: ???
```

79. ¿Cuáles son las diferencias entre la ejecución del programa `"Hola Mundo"` con el intérprete interactivo y con PRY? Escriba la respuesta en el fichero creado en el ejercicio 18.

80. Añada todo el contenido del directorio al *índice del repositorio git*. (`git add .`)

81. Confirme los cambios. (`git commit -m "Creado byebyeWorld.rb para usar con PRY"`)

82. Muestre las confirmaciones realizadas en el repositorio hasta el momento.
    (`git log --graph --abbrev-commit --decorate --date=relative --all`)

83. Empuje los cambios en el repositorio remoto denominado *ghp03*. (`git push -u ghp03 master`)

84. Escriba la dirección HTTP del repositorio de la organización 'ULL-ESIT-LPP-1920' en la tarea habilitada en el campus virtual. (`https://github.com/ULL-ESIT-LPP-1920/rebase-aluXXX.git`)

85. Cierre la sesión.