

Федеральное государственное автономное образовательное учреждение высшего образования «**Национальный исследовательский университет ИТМО**»

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №2
по дисциплине «**Основы программной инженерии**»

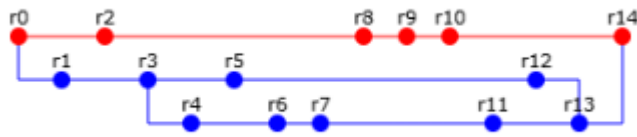
Вариант: **16095**

Преподаватель:

Выполнил: Садовой Григорий Владимирович
Группа: P3207

Санкт-Петербург, 2025

1. Задание



Сконфигурировать в своём домашнем каталоге репозитории svn и git и загрузить в них начальную ревизию файлов с исходными кодами (в соответствии с выданным вариантом).

Воспроизвести последовательность команд для систем контроля версий svn и git, осуществляющих операции над исходным кодом, приведённые на блок-схеме.

При составлении последовательности команд необходимо учитывать следующие условия:

- Цвет элементов схемы указывает на пользователя, совершившего действие (красный - первый, синий - второй).
- Цифры над узлами - номер ревизии. Ревизии создаются последовательно.
- Необходимо разрешать конфликты между версиями, если они возникают.

2. Реализация с использованием Git

```
1. echo "====="
2. echo "Скрипт для лабораторной работы номер 2 по ОПИ."
3. echo "Скрипт для git."
4. echo -e "=====\n"
5.
6. echo "Подготовка директорий."
7. ./delete.sh
8. echo -e "=====\n"
9.
10. echo "Инициализация репозитория."
11. git init
12. echo -e "=====\n"
13.
14. echo -e "Настраиваем редактор по умолчанию для merge tool.\n"
15. echo -e "\n[merge]\n\tool = vscode" >> .git/config
16.
17. echo "Создание основного пользователя(красный)."
18. git config user.name "red"
19. git config user.email "red@mail.ru"
20. echo "Основной пользователь создан."
21. echo -e "=====\n"
22.
23. git checkout -b branch1
24.
25. echo "commits" >> .gitignore
26. echo "src/.gitkeep" >> .gitignore
27. echo ".editorconfig" >> .gitignore
28. echo "git.sh" >> .gitignore
29. echo ".gitattributes" >> .gitignore
30. echo "delete.sh" >> .gitignore
```

```
31.git add .gitignore
32.echo "Новый .gitignore создан"
33.echo -e "=====\n"
34.

35.unzip -o ../commits/commit0.zip -d src
36.git add .
37.git commit -m "Initial first commit (r0)"
38.echo "Коммит r0 создан."
39.echo -e "=====\n"
40.
41.git checkout -b branch2
42.
43.unzip -o ../commits/commit1.zip -d src
44.git add .
45.git commit --author="blue <blue@mail.ru>" -m "Commit r1"
46.echo "Коммит r1 создан."
47.
48.git checkout branch1
49.
50.unzip -o ../commits/commit2.zip -d src
51.git add .
52.git commit -m "Commit r2"
53.echo "Коммит r2 создан."
54.echo -e "=====\n"
55.
56.git checkout branch2
57.
58.unzip -o ../commits/commit3.zip -d src
59.git add .
60.git commit --author="blue <blue@mail.ru>" -m "Commit r3"
61.echo "Коммит r3 создан."
62.echo -e "=====\n"
63.
64.git checkout -b branch3
65.
66.unzip -o ../commits/commit4.zip -d src
67.git add .
68.git commit --author="blue <blue@mail.ru>" -m "Commit r4"
69.echo "Коммит r4 создан."
70.echo -e "=====\n"
71.
72.git checkout branch2
73.
74.unzip -o ../commits/commit5.zip -d src
75.git add .
76.git commit --author="blue <blue@mail.ru>" -m "Commit r5"
77.echo "Коммит r5 создан."
78.echo -e "=====\n"
79.
80.git checkout branch3
81.
82.unzip -o ../commits/commit6.zip -d src
```

```
83.git add .
84.git commit --author="blue <blue@mail.ru>" -m "Commit r6"
85.echo "Коммит r6 создан."
86.echo -e "=====\n"
87.
88.unzip -o ../commits/commit7.zip -d src
89.git add .
90.git commit --author="blue <blue@mail.ru>" -m "Commit r7"
91.echo "Коммит r7 создан."
92.echo -e "=====\n"
93.
94.git checkout branch1
95.
96.unzip -o ../commits/commit8.zip -d src
97.git add .
98.git commit -m "Commit r8"
99.echo "Коммит r8 создан."
100.    echo -e "=====\n"
101.
102.    unzip -o ../commits/commit9.zip -d src
103.    git add .
104.    git commit -m "Commit r9"
105.    echo "Коммит r9 создан."
106.    echo -e "=====\n"
107.
108.    unzip -o ../commits/commit10.zip -d src
109.    git add .
110.    git commit -m "Commit r10"
111.    echo "Коммит r10 создан."
112.    echo -e "=====\n"
113.
114.    git checkout branch3
115.
116.    unzip -o ../commits/commit11.zip -d src
117.    git add .
118.    git commit --author="blue <blue@mail.ru>" -m "Commit r11"
119.    echo "Коммит r11 создан."
120.    echo -e "=====\n"
121.
122.    git checkout branch2
123.
124.    unzip -o ../commits/commit12.zip -d src
125.    git add .
126.    git commit --author="blue <blue@mail.ru>" -m "Commit r12"
127.    echo "Коммит r12 создан."
128.    echo -e "=====\n"
129.
130.    git checkout branch3
131.
```

```

132.      #3 варианта:
133.
134.      # 1 вариант: своими руками.
135.      # git merge --no-commit branch2
136.      # git mergetool
137.      # git commit --author="blue <blue@mail.ru>" -m "Слияние 2 и 3 ветки
    голубым"
138.      # git merge --abort (если хотим отменить слияние)
139.
140.      # 2 вариант: сохранение ветки из которой вызывают.
141.      git merge --no-commit branch2 -Xours
142.      git commit --author="blue <blue@mail.ru>" -m "Слияние 2 и 3 ветки
    голубым"
143.
144.      # 3 вариант: сохранение ветки которую вызывают.
145.      # git merge --no-commit branch2 -Xtheirs
146.      # git commit --author="blue <blue@mail.ru>" -m "Слияние 2 и 3 ветки
    голубым"
147.
148.      git checkout branch1
149.      git merge --no-commit branch2 -Xours
150.      git commit -m "Слияние 3 и 1 ветки красным"
151.
152.      git log --graph --all --oneline
153.
154.      git branch -vv
155.
156.      git branch -d branch2
157.
158.      git branch -vv
159.
160.

```

3. Реализация с использованием SVN

```

1. echo "=====
2. echo "Скрипт для лабораторной работы номер 2 по ОПИ."
3. echo "Скрипт для svn."
4. echo -e "=====\n"
5.
6. echo "Очистка предыдущих данных..."
7. rm -rf test_rep work_rep
8. echo -e "=====\n"
9.
10. echo "Создание нового репозитория..."
11. svnadmin create test_rep
12. REPO_URL="file://$(pwd)/test_rep"
13. echo -e "=====\n"

```

```
14.
15. echo "Создание структуры репозитория..."
16. svn mkdir -m "Создаем стандартную структуру." \
17.     "$REPO_URL/trunk" \
18.     "$REPO_URL/branches" \
19.     "$REPO_URL/tags"
20. echo -e "=====\n"
21.
22. echo "Создание рабочей копии..."
23. svn checkout "$REPO_URL/trunk" work_rep
24. cd work_rep
25. echo -e "=====\n"
26.
27. svn resolve --accept=postpone -R .
28.
29. echo "Добавление начальных файлов..."
30. unzip -o ../../commits/commit0.zip -d .
31. svn add --force .
32. svn commit -m "Initial first commit r0" --username red
33. echo "Коммит r0 создан."
34. echo -e "=====\n"
35.
36. svn copy $REPO_URL/trunk $REPO_URL/branches/branch2 -m "Creating branch2"
37. svn switch $REPO_URL/branches/branch2
38.
39. svn rm *
40. unzip -o ../../commits/commit1.zip -d .
41. svn add *
42. svn commit -m "Commit r1" --username=blue
43. echo "Коммит r1 создан."
44. echo -e "=====\n"
45.
46. svn switch $REPO_URL/trunk
47. svn rm *
48. unzip -o ../../commits/commit2.zip -d .
49. svn add *
50. svn commit -m "Commit r2" --username=red
51. echo "Коммит r2 создан."
52. echo -e "=====\n"
53.
54. svn switch $REPO_URL/branches/branch2
55. svn rm *
56. unzip -o ../../commits/commit3.zip -d .
57. svn add *
58. svn commit -m "Commit r3" --username=blue
59. echo "Коммит r3 создан."
60. echo -e "=====\n"
61.
62. svn copy $REPO_URL/branches/branch2 $REPO_URL/branches/branch3 -m "Creating
    branch3"
63. svn switch $REPO_URL/branches/branch3
64.
```

```
65.svn rm *
66.unzip -o ../../commits/commit4.zip -d .
67.svn add *
68.svn commit -m "Commit r4" --username=blue
69.echo "Коммит r4 создан."
70.echo -e "=====\n"
71.
72.svn switch $REPO_URL/branches/branch2
73.svn rm *
74.unzip -o ../../commits/commit5.zip -d .
75.svn add *
76.svn commit -m "Commit r5" --username=blue
77.echo "Коммит r5 создан."
78.echo -e "=====\n"
79.

80.svn switch $REPO_URL/branches/branch3
81.svn rm *
82.unzip -o ../../commits/commit6.zip -d .
83.svn add *
84.svn commit -m "Commit r6" --username=blue
85.echo "Коммит r6 создан."
86.echo -e "=====\n"
87.

88.svn rm *
89.unzip -o ../../commits/commit7.zip -d .
90.svn add *
91.svn commit -m "Commit r7" --username=blue
92.echo "Коммит r7 создан."
93.echo -e "=====\n"
94.

95.svn switch $REPO_URL/trunk
96.svn rm *
97.unzip -o ../../commits/commit8.zip -d .
98.svn add *
99.svn commit -m "Commit r8" --username=red
100.    echo "Коммит r8 создан."
101.    echo -e "=====\n"
102.
103.    svn rm *
104.    unzip -o ../../commits/commit9.zip -d .
105.    svn add *
106.    svn commit -m "Commit r9" --username=red
107.    echo "Коммит r9 создан."
108.    echo -e "=====\n"
109.

110.    svn rm *
111.    unzip -o ../../commits/commit10.zip -d .
```


```
112.     svn add *
113.     svn commit -m "Commit r10" --username=red
114.     echo "Коммит r10 создан."
115.     echo -e "=====\n"
116.
117.     svn switch $REPO_URL/branches/branch3
118.     svn rm *
119.     unzip -o ../../commits/commit11.zip -d .
120.     svn add *
121.     svn commit -m "Commit r11" --username=blue
122.     echo "Коммит r11 создан."
123.     echo -e "=====\n"
124.
125.     svn switch $REPO_URL/branches/branch2
126.     svn rm *
127.     unzip -o ../../commits/commit12.zip -d .
128.     svn add *
129.     svn commit -m "Commit r12" --username=blue
130.     echo "Коммит r12 создан."
131.     echo -e "=====\n"
132.
133.     svn update
134.
135.     svn switch $REPO_URL/branches/branch3
136.     svn merge $REPO_URL/branches/branch2
137.
138.     svn resolve --accept=working D.java
139.     svn resolve --accept=working H.java
140.     svn resolve --accept=working I.java
141.     svn resolve --accept=working Eq1Tgt0j2T.731
142.
143.     svn commit -m "Слияние 2 и 3 ветки голубым пользователем." --
        username=blue
144.
145.     # Или еще варианты
146.     #
147.     #
148.     # Через svn resolve --accept=theirs-full D.java
149.     #
150.     #
151.     # Или вручную
152.     # svn cat H.java > H.java.mine
153.     # svn cat ^/branches/branch2/H.java@HEAD > H.java.theirs
154.     # code -d H.java.mine H.java.theirs
155.     # code H.java сохраняем изменения
156.     # svn resolve --accept=working H.java
157.     #
```




```
158.      # svn cat I.java > I.java.mine
159.      # svn cat ^/branches/branch2/I.java@HEAD > I.java.theirs
160.      # code -d I.java.mine I.java.theirs
161.      # code I.java
162.      # svn resolve --accept=working I.java
163.      #
164.      # svn resolve --accept=working Eq1Tgt0j2T.73l
165.      # или
166.      # svn cp ^/branches/branch2/Eq1Tgt0j2T.73l@HEAD Eq1Tgt0j2T.73l
167.      # svn resolve --accept=working Eq1Tgt0j2T.73l
168.      #
169.      # svn commit -m "Ручное слияние ветки 2 и ветки 3" --username=blue
170.      #
171.      #
172.
173.      svn rm *
174.      unzip -o ../../commits/commit13.zip -d .
175.      svn add *
176.      svn commit -m "Commit r13" --username=blue
177.      echo "Коммит r13 создан."
178.      echo -e "=====\n"
179.
180.      svn switch $REPO_URL/trunk
181.      svn merge $REPO_URL/branches/branch3
182.
183.      svn resolve --accept=working D.java
184.      svn resolve --accept=working H.java
185.      svn resolve --accept=working I.java
186.
187.      svn commit -m "Слияние 1 и 3 ветки красным пользователем." --
        username=red
188.
189.      svn rm *
190.      unzip -o ../../commits/commit14.zip -d .
191.      svn add *
192.      svn commit -m "Commit r14" --username=blue
193.      echo "Коммит r14 создан."
194.      echo -e "=====\n"
195.
196.
```

4. Вывод

В ходе выполнения лабораторной работы я ознакомился с основами Git и Subversion (SVN). В рамках задания я настроил SVN- и Git-репозитории в домашней директории, загрузил первоначальные версии файлов с исходным кодом и выполнил необходимые операции в соответствии с требованиями. Я освоил базовые команды обеих систем, а также научился устранять конфликты при слиянии изменений.

 Варить товар с Джесси

 Комментировать MLBB