Repozytorium kodu GitHub – tworzenie

1. W przeglądarce otwórz stronę: https://github.com/
2. Zaloguj się, lub utwórz konto w systemie
3. W **AndroidStudio** utwórz nowy projekt:
   - Nazwa taka sama jak nazwa repozytorium np. „android-lab”
   - API 22: Android 5.1 (Lollipop)
   - Empty Activity
4. Poczekaj aż „Gradle” zakończy pracę
5. Z górnego menu wybierz opcje: VCS->Import into Version Control -> Share Project on GitHub
6. W oknie logowania odznacz opcję „Save password”
7. Wprowadź swój login i hasło
8. Wprowadź nazwę i opis repozytorium (projektu)
9. Z okna wyboru plików do wersjonowania wybierz tylko katalog „main” (app->C:/Users/…->src->main)
10. Sprawdź na portalu https://github.com/ czy repozytorium zostało utworzone i jakie pliki zawiera

GIT jest rozproszonym systemem wersjonowania kodu, pozwalający wielu ludziom pracować nad jednym projektem.

W odróżnieniu od SVN, wprowadzane zmiany w kodzie są zapisywane na dyskach lokalnych poszczególnych programistów, którzy je opracowali, a dopiero później wysyłane na wspólny serwer. Dzięki temu bez dostępu do sieci mogą oni kontynuować pracę i czerpać korzyści z wersjonowania plików.

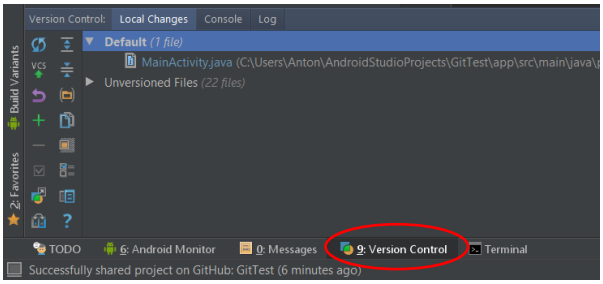
SVN pozwala na pobranie poszczególnych plików – GIT jedynie całego projektu

GIT wprowadza uprawnienia do zarządzania wersjonowaniem – tylko właściciel może zatwierdzić zmiany w projekcie głównym.

GIT przy dużych projektach działa szybciej od SVN i zajmuje mniej miejsca na dysku

Repozytorium kodu GitHub – Wersjonowanie i cofanie wprowadzonych zmian

1. W pliku MainActivity.java wprowadź jakieś zmiany (np. komentarz *TODO*)
2. Wejdź w zakładkę Version Control i sprawdź jakie pliki zostały zmienione



W zakładce Version Control, która pojawiła się po podpięciu repozytorium możliwe jest sterowanie wersjonowaniem.

W zakładce automatycznie pokazane są pliki w których wprowadzono jakieś zmiany w stosunku do ostatniego „commita”

3. Aby zatwierdzić zmiany musisz wykonać „commit”, w tym celu kliknij „Commit changes” (ikonka zielonej strzałki VCS, z poziomu menu VCS, lub skrót klawiaturowy ctrl+K
4. Wprowadź wiadomość opisującą danego „commita”
5. Sprawdź czy commit dodał się w

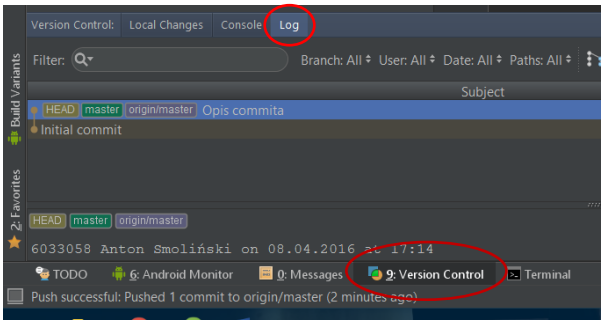
Każde utworzenie Commita powoduje utworzenie wersji projektu do której można cofnąć wprowadzone zmiany.

systemie https://github.com/

6. Aby wgrać commita na serwer GitHub musisz wykonać „Push commit” (np. z poziomu menu VCS)
7. Sprawdź, czy zmiany zostały zapisane na serwerze GitHub
8. Aby przywrócić poprzedni plik (poprzedniego commita) w zakładce Version Control przejdź do podzakładki LOG

Commit jest przechowywany lokalnie na komputerze gdzie został utworzony.

Aby stworzyć kopię Commita na zewnętrznym serwerze (GitHub) należy użyć opcji Push



GitHub pozwala na tworzenie nowych „gałęzi” (branches) projektu i rozwijanie ich równolegle – wówczas należy wybrać do której gałęzi zapisać dane zmiany. Z reguły tylko właściciel projektu ma możliwość dodawania zmian w gałęzi master, pozostali użytkownicy wprowadzający zmianę (commit) tworzą własne gałęzie projektu. Właściciel może scalić dane gałęzie jeżeli uzna, że dane zmiany są istotne.

9. Z listy commitów wybierz tego którego chcesz cofnąć (czyli tego ostatniego)
10. Z prawej strony okna znajdziesz listę plików zmienionych w danym commicie, nad nimi znajduje się fioletowa strzałka „Revert Selected Changes”
11. Po kliknięciu na strzałkę w pliku MainActivity.java powinien zniknąć dodany komentarz

W pracy grupowej ważną opcją jest także „Update Project” z menu VCS, która pozwala na pobranie aktualnych Commitów projektu z serwera GitHub (wszystkich użytkowników pracujących nad danym projektem)

Za pomocą opcji Pull można pobrać Commit z serwera GitHub do lokalnego repozytorium (np. commit dokonany przez innego użytkownika)

Tworzenie listy wpisów

1. W layoucie activity_main.xml utwórz kontrolkę ListView (dział Containers)
2. Usuń kontrolkę TextView i ustaw ListView na cały obszar aplikacji
3. Utwórz 2 zmienne prywatne klasy MainActivity:

Lista jest dynamiczną strukturą, która wyświetla tyle obiektów ile zostanie do niej przekazane.

```
private ArrayList<String> target;
private ArrayAdapter adapter;
```

```
String[] values = new String[] { "Pies",
"Kot", "Koń", "Gołąb", "Kruk", "Dzik", "Karp",
"Osioł", "Chomik", "Mysz", "Jeż", "Kraluch" };
```

4. W pliku MainActivity.java w metodzie onCreate utwórz tablicę typu String, która będzie zawierać elementy wyświetlane na liście
5. Umieść elementy z tablicy values (utworzonej w pkt 4) do struktury ArrayList o nazwie target:

Elementy wyświetlane przez listę muszą zostać umieszczone w strukturze o nazwie Adapter np. ArrayAdapter

```
this.target = new ArrayList<String>();
this.target.addAll(Arrays.asList(values));
```

Struktura ArrayList pozwala na łatwe dodawanie i usuwanie elementów które się w niej znajdują

6. Zainicjalizuj zmienną this.adapter

```
this.adapter = new ArrayAdapter(this,
   android.R.layout.simple_list_item_1,
```

```
    this.target);
```

7. Zmapuj kontrolkę ListView i użyj na niej metody .setAdapter(adapter) aby ustawić adapter do listy
8. Uruchom aplikację i sprawdź, czy lista została wyświetlona prawidłowo.
9. Przeprowadź Commita

```
ListView listview = (ListView) findViewById(
R.id.listView );
listview.setAdapter(this.adapter);
```

Tworzenie formularza dodawania wpisu

1. Kliknij PPM na folder „res” na drzewie struktury projektu
2. Wybierz opcję New->Android resource directory
3. W formularzu wybierz „Resource type” – menu i kliknij ok
4. W nowo utworzonym folderze „menu” utwórz plik (New ->Menu resource file) o nazwie main_menu.xml
5. Dodaj w utworzonym pliku element item:

```
<item
  android:id="@+id/create"
  android:title="Nowy"
  android:onClick="nowyWpis"
  android:icon="@android:drawable/ic_menu_add"
  app:showAsAction="always"
  />
```

6. Do tagu <menu> dodaj parametr:

```
xmlns:app="http://schemas.android.com/apk/res-auto"
```

7. Przeciąż w pliku MainActivity.java metodę onCreateOptionsMenu, która doda zasób main_menu do danego activity

```
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.main_menu, menu);
    return true;
}
```

8. Utwórz w publiczną metodę void nowyWpis przyjmującą jako parametr obiekt typu MenuItem
9. W metodzie „nowyWpis” dodaj wywołanie intencji:

```
public void nowyWpis(MenuItem mi)
{
    Intent intencja = new Intent(this,
DodajWpis.class);
    startActivityForResult(intencja, 1);
}
```

10. Stwórz nową aktywność (new->Activity->EmptyActivity) o nazwie „DodajWpis”

Formularz nowego wpisu będzie dostępny w nowym activity uruchamianym z menu aplikacji.

Na początku należy utworzyć menu – zasób xml, a następnie wywołać ładowanie tego zasobu z Activity

Parametry:

- *Id- identyfikuje opcje,*
- *Tile – wyświetlana nazwa*
- *onClick - metoda wywoływana po kliknięciu na daną opcję*
- *Icon – ikonka opcji*
- *showAsAction – określa sposób wyświetlania opcji, w tym przypadku „always” oznacza że opcja będzie wyświetlana na pasku zadań a nie w rozwijanym menu*

Uwaga, nie mamy jeszcze utworzonej metody nowyWpis – zostanie utworzona w pkt 8.

Aby przypisać metodę do parametru onClick musi ona być public void, i zawierać 1 parametr

- *View – dla kontrolek dziedziczących po View (np. Button)*
- *MenuItem – dla opcji w menu*

W metodzie uruchamiamy nieistniejącą jeszcze Activity DodajWpis, dlatego IDE podkreśla tą nazwę.

Finkcja startActivityForResult() uruchamia intencję w oczekiwaniu na to, że coś otrzyma po jej wykonaniu, w tym celu należy nadać i podać kod wywołania danej intencji liczbami naturalnymi (by móc zidentyfikować odebrane dane)

11. W nowej aktywności (DodajWpis) dodaj dwie kontrolki – PlainText (TextField), oraz Button (Widgets)
12. Przeprowadź Commita

W nowym Activity ustalamy wygląd formularza dodawania opcji

Obsługa formularza dodawania

1. W pliku DodajWpis.java utwórz publiczną metodę void „wyślij” przyjmującą jako parametr obiekt typu View
2. Zmapuj kontrolkę EditText
3. Na zmapowanym obiekcie uruchom metodę getText().toString(), która zwróci zmienną typu String, którą wprowadził użytkownik
4. Utwórz obiekt intencji
5. Na obiekcie intencji uruchom metodę .putExtra("wpis", pole) która utworzy zmienną „wpis” o wartości jaka znajduje się zmiennej pole
6. Wywołaj metodę setResult(RESULT_OK, intencja), która spowoduje, że Activity jako wynik swojej pracy zwróci obiekt typu intencja
7. Zakończ działanie Activity funkcją finish()
8. Przypisz metodę „wyslij” do Buttona (właściwość onClick)
9. W pliku MainActivity.java przeciąż metodę onActivityResult, która odbierze dane z formularza i doda je do adaptera
10. Przeprowadź Commita i push-a

Aby mapować obiekt należy utworzyć obiekt żądanego typu, a następnie przypisać do niego rzutowany na dany typ wynik funkcji findViewById(R.id.id_kontrolki)

```
public void wyslij (View view)
{
     EditText kontrolka =
(EditText)findViewById(R.id.editText);
    String pole =
kontrolka.getText().toString();
    Intent intencja = new Intent();
    intencja.putExtra("wpis", pole);
    setResult(RESULT_OK, intencja);
    finish();
}
```

Intencje służą nie tylko do wywoływania nowych aktywności, ale także do komunikacji między nimi

```
@Override
protected void onActivityResult(
int requestCode, int resultCode, Intent data)
{
  if(requestCode==1 && resultCode==RESULT_OK)
  {
     Bundle extras = data.getExtras();
     String nowy = (String)extras.get("wpis");
     target.add(nowy);
     adapter.notifyDataSetChanged();
    }
}
```