Pets part 2

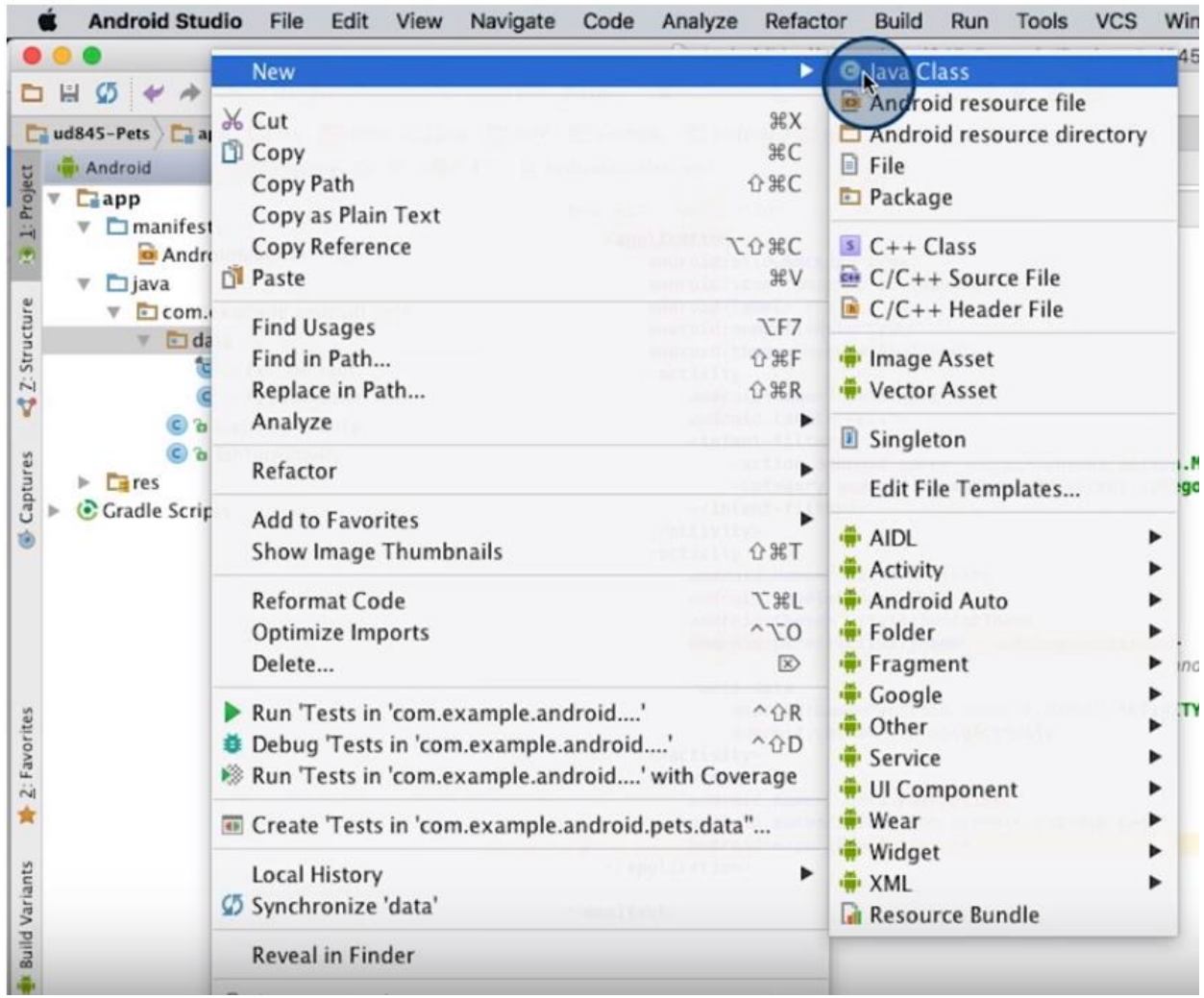
07 - Add ContentProvider

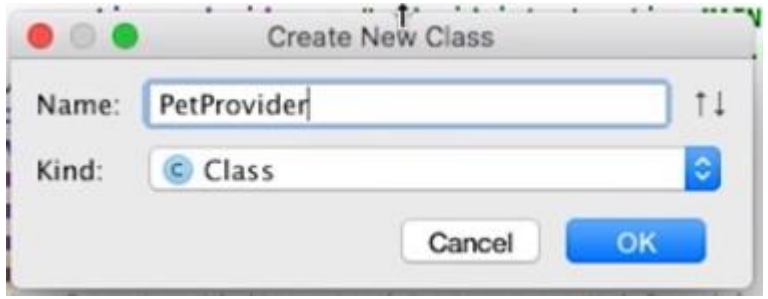
Copy PetProvider Class from:
https://gist.github.com/udacityandroid/7cf842c9f191f89559c333ef895bc415

ADD CONTENT PROVIDER TO PETS APP

- [ ] Create PetProvider class:
  - Copy over contents from gist linked below into PetProvider.java file
  - Finish the TODO in that file
- [ ] Declare PetProvider as an application component of the Pets app in AndroidManifest.xml:
  - Use the <provider> XML tag

Create PetProvider.java





```
        </activity>
        <provider
            android:name=".data.PetProvider"
            android:authorities="com.example.android.pets"
            android:exported="false" />
    </application>

</manifest>
```

```
<provider
  android:name=".data.PetProvider"
  android:authorities="com.example.android.pets"
  android:exported="false" />
```

AndroidManifest.xml

08 - Add constants for pet content URIs

Modify PetContract.java

```
public final class PetContract {
    private PetContract (){
    }
    public static final String CONTENT_AUTHORITY = "com.example.android.pets";
    public static final Uri BASE_CONTENT_URI = Uri.parse("content://" + CONTENT_AUTHORITY);
    public static final String PATH_PETS = "pets";

    public static final class PetEntry implements BaseColumns{
        public static final Uri CONTENT_URI = Uri.withAppendedPath(BASE_CONTENT_URI, PATH_PETS);

        public static final String TABLE_NAME = "pets";

        public static final String _ID = BaseColumns._ID;
        public static final String COLUMN_PET_NAME = "name";
        public static final String COLUMN_PET_BREED = "breed";
        public static final String COLUMN_PET_GENDER = "gender";
        public static final String COLUMN_PET_WEIGHT = "weight";
```

09 - Add UriMatcher to ContentProvider PetProvider.java

```
public class PetProvider extends ContentProvider {

  private static final int PETS = 100;
  private static final int PET_ID = 101;

  private static final UriMatcher sUriMatcher = new UriMatcher(UriMatcher.NO_MATCH);

  static {

    sUriMatcher.addURI( PetContract.CONTENT_AUTHORITY, PetContract.PATH_PETS , PETS );

    sUriMatcher.addURI( PetContract.CONTENT_AUTHORITY, PetContract.PATH_PETS + "/#", PETS_ID );
  }

  ...
}
```

10 - Implement ContentProvider query() method

IMPLEMENT CONTENT PROVIDER QUERY() METHOD

- [ ] 1. Replace query() method in PetProvider class with the code gist linked below.
- [ ] 2. Finish the TODO in the PETS case to perform a query on the database with the given projection, selection, selection args, and sort order.
- [ ] 3. Ensure the app still compiles. No visible change to the app is expected.

Modify PetProvider.java

- Copy query() method from this link
- https://gist.github.com/udacityandroid/ea56537b59ad44504aeb1688727189dd
- Paste at query()

```
@Override
public Cursor query(Uri uri, String[] projection, String selection, String[] selectionArgs,
                    String sortOrder) {
    return null;
}
```

PetProvider.java

Add variable (put before onCreate)

```
/** Database helper object */
private PetDbHelper mDbHelper;
```

Modify Query (case PETS)

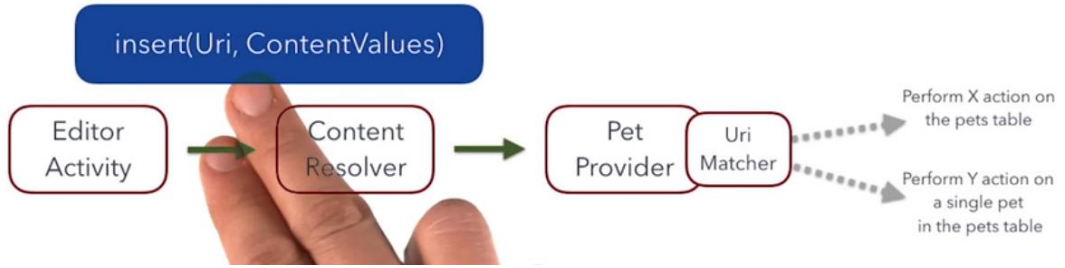
```
public Cursor query(Uri uri, String[] projection, String selection, String[] selectionArgs,
                    String sortOrder) {
    // Get readable database
    SQLiteDatabase database = mDbHelper.getReadableDatabase();

    // This cursor will hold the result of the query
    Cursor cursor;

    // Figure out if the URI matcher can match the URI to a specific code
    int match = sUriMatcher.match(uri);
    switch (match) {
        case PETS:
            // For the PETS code, query the pets table directly with the given
            // projection, selection, selection arguments, and sort order. The cursor
            // could contain multiple rows of the pets table.
            cursor = database.query(PetEntry.TABLE_NAME, projection, selection, selectionArgs,
                    null, null, sortOrder);
            break;
```

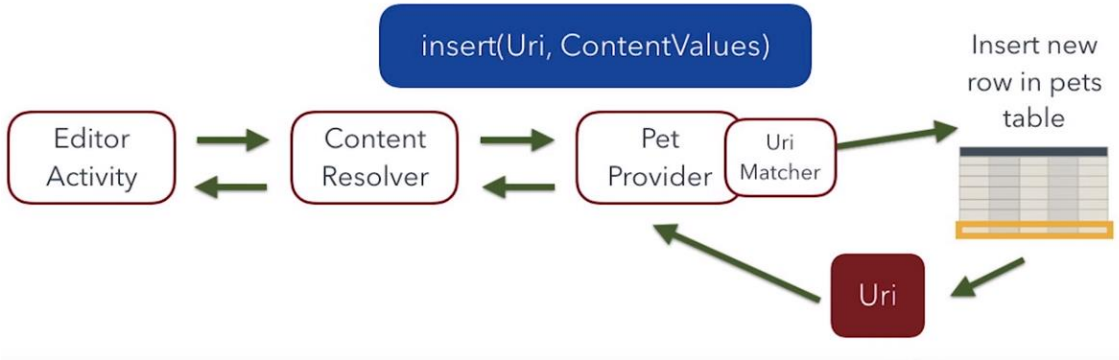
10 - Implement ContentProvider query() method modify displayDatabaseInfo()

```java
private void displayDatabaseInfo(){

    SQLiteDatabase db = mDbHelper.getReadableDatabase();

    String[] projection = {
            PetContract.PetEntry._ID,
            PetContract.PetEntry.COLUMN_PET_NAME,
            PetContract.PetEntry.COLUMN_PET_BREED,
            PetContract.PetEntry.COLUMN_PET_GENDER,
            PetContract.PetEntry.COLUMN_PET_WEIGHT
    };

    Cursor cursor = db.query(
            PetContract.PetEntry.TABLE_NAME,
            projection,
            selection: null,
            selectionArgs: null,
            groupBy: null,
            having: null,
            orderBy: null
    );
    TextView displayView = (TextView)findViewById(R.id.text_view_pet);
```

```java
private void displayDatabaseInfo() {
    // Define a projection that specifies which columns from the database
    // you will actually use after this query.
    String[] projection = {
            PetEntry._ID,
            PetEntry.COLUMN_PET_NAME,
            PetEntry.COLUMN_PET_BREED,
            PetEntry.COLUMN_PET_GENDER,
            PetEntry.COLUMN_PET_WEIGHT };

    // Perform a query on the provider using the ContentResolver.
    // Use the {@link PetEntry#CONTENT_URI} to access the pet data.
    Cursor cursor = getContentResolver().query(
            PetEntry.CONTENT_URI,   // The content URI of the words table
            projection,             // The columns to return for each row
            null,                   // Selection criteria
            null,                   // Selection criteria
            null);                  // The sort order for the returned rows

    TextView displayView = (TextView) findViewById(R.id.text_view_pet);
```

11 - Query the provider using the pet content URI (PetProvider.java)





PET PROVIDER INSERT METHOD

URIMatcher

Can only insert into pets table

PETS case

Not supported

~~PET_ID case~~

Get database object

Insert pet

PetProvider.java

Modify Uri insert () method

```
@Override
public Uri insert(Uri uri, ContentValues contentValues) {
    final int match = sUriMatcher.match(uri);
    switch (match) {
        case PETS:
            return insertPet(uri, contentValues);
        default:
            throw new IllegalArgumentException("Insertion is not supported for " + uri);
    }
}
```

In PetProvider: Create Uri insertPet() method

```
import android.util.Log;
```

```
private Uri insertPet(Uri uri, ContentValues values) {
    // Get writeable database
    SQLiteDatabase database = mDbHelper.getWritableDatabase();

    // Insert the new pet with the given values
    long id = database.insert(PetEntry.TABLE_NAME, null, values);
    // If the ID is -1, then the insertion failed. Log an error and return null.
    if (id == -1) {
        Log.e(LOG_TAG, "Failed to insert row for " + uri);
        return null;
    }

    // Return the new URI with the ID (of the newly inserted row) appended at the end
    return ContentUris.withAppendedId(uri, id);
}
```

In PetProvider: Modify onCreate()

```
@Override
public boolean onCreate() {
    mDbHelper = new PetDbHelper(getContext());
    return true;
}
```

12 - Implement and use ContentProvider insert() method

- Use PetProvider insert () methd in Catalaog Actvity
  - Remove from onCreate()

```
mDbHelper = new PetDbHelper( context: this);
displayDatabaseInfo();
```

- Use PetProvider insert () methd in Editor Actvity
  - Modify variable mGender to:

```
private int mGender = PetEntry.GENDER_UNKNOWN;

public void onNothingSelected(AdapterView<?> parent) {
    mGender = PetEntry.GENDER_UNKNOWN;
}
```

Use PetProvider insert() method

- In Catalog Activity inserPet()

```
private void insertPet(){

    ContentValues values = new ContentValues();
    values.put(PetContract.PetEntry.COLUMN_PET_NAME, "Toto");
    values.put(PetContract.PetEntry.COLUMN_PET_BREED, "Poemeranian");
    values.put(PetContract.PetEntry.COLUMN_PET_GENDER, PetContract.PetEntry.GENDER_MALE);
    values.put(PetContract.PetEntry.COLUMN_PET_WEIGHT, 10);

    Uri newUri = getContentResolver().insert(PetContract.PetEntry.CONTENT_URI, values);

}
```

In EditorActivity.java: insertPet()

```
private void insertPet(){
    String nameString = mNameEditText.getText().toString().trim();
    String breedString = mBreedEditText.getText().toString().trim();
    String weightString = mWeightEditText.getText().toString().trim();
    int weight = Integer.parseInt(weightString);

    ContentValues values = new ContentValues();
    values.put(PetContract.PetEntry.COLUMN_PET_NAME, nameString);
    values.put(PetContract.PetEntry.COLUMN_PET_BREED, breedString);
    values.put(PetContract.PetEntry.COLUMN_PET_GENDER, mGender);
    values.put(PetContract.PetEntry.COLUMN_PET_WEIGHT, weight);

    Uri newUri = getContentResolver().insert(PetEntry.CONTENT_URI, values);

    if (newUri == null){
        Toast.makeText( context: this, getString(R.string.editor_insert_pet_failed),
                Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, getString(R.string.editor_insert_pet_successful),
                Toast.LENGTH_SHORT).show();
    }
}
```

In res/values/strings.xml:

```
<!-- Toast message in editor when new pet has been successfully inserted [CHAR LIMIT=NONE] -->
<string name="editor_insert_pet_successful">Pet saved</string>

<!-- Toast message in editor when new pet has failed to be inserted [CHAR LIMIT=NONE] -->
<string name="editor_insert_pet_failed">Error with saving pet</string>
```

Run the app

- Run the app and test that creating a new pet still works correctly. If so, congratulations! You implemented insert() functionality in the ContentProvider and updated the UI code to call into that provider code!
- Try Insert Dummy Data
- Try Insert Pet from Editor

13 - Add input validation to ContentProvider insert() method

- **Add sanity checks to PetProvider insert() and update() methods**
- **Check the values in the ContentValues object**
- **Step 1: Determine requirements for each type of data**
- **Step 2: Add checks in the code to enforce these requirements**

Modify PetContract.java (add isValidGender

```
public static boolean isValidGender(int gender) {
    if (gender == GENDER_UNKNOWN || gender == GENDER_MALE || gender == GENDER_FEMALE) {
        return true;
    }
    return false;
}
```

In PetProvider:

Modify Uri insertPet by Adding data validation

```
private Uri insertPet(Uri uri, ContentValues values) {
    // Check that the name is not null
    String name = values.getAsString(PetEntry.COLUMN_PET_NAME);
    if (name == null) {
        throw new IllegalArgumentException("Pet requires a name");
    }

    // Check that the gender is valid
    Integer gender = values.getAsInteger(PetEntry.COLUMN_PET_GENDER);
    if (gender == null || !PetEntry.isValidGender(gender)) {
        throw new IllegalArgumentException("Pet requires valid gender");
    }

    // If the weight is provided, check that it's greater than or equal to 0 kg
    Integer weight = values.getAsInteger(PetEntry.COLUMN_PET_WEIGHT);
    if (weight != null && weight < 0) {
        throw new IllegalArgumentException("Pet requires valid weight");
    }

    // No need to check the breed, any value is valid (including null).

    // Get writeable database
    SQLiteDatabase database = mDbHelper.getWritableDatabase();
```

14 - Implement ContentProvider update() method

- Modify update () in PetProvider

```java
@Override
public int update(Uri uri, ContentValues contentValues, String selection,
                  String[] selectionArgs) {
    final int match = sUriMatcher.match(uri);
    switch (match) {
        case PETS:
            return updatePet(uri, contentValues, selection, selectionArgs);
        case PET_ID:
            // For the PET_ID code, extract out the ID from the URI,
            // so we know which row to update. Selection will be "_id=?" and selection
            // arguments will be a String array containing the actual ID.
            selection = PetEntry._ID + "=?";
            selectionArgs = new String[] { String.valueOf(ContentUris.parseId(uri)) };
            return updatePet(uri, contentValues, selection, selectionArgs);
        default:
            throw new IllegalArgumentException("Update is not supported for " + uri);
    }
}
```

Create updatePet method in PetProvider

```
private int updatePet(Uri uri, ContentValues values, String selection, String[] selectionArgs) {

    // TODO: Update the selected pets in the pets database table with the given ContentValues

    // TODO: Return the number of rows that were affected
    return 0;
}
```

Modify updatePet()

```
private int updatePet(Uri uri, ContentValues values, String selection, String[] selectionArgs) {
    // If the {@link PetEntry#COLUMN_PET_NAME} key is present,
    // check that the name value is not null.
    if (values.containsKey(PetEntry.COLUMN_PET_NAME)) {
        String name = values.getAsString(PetEntry.COLUMN_PET_NAME);
        if (name == null) {
            throw new IllegalArgumentException("Pet requires a name");
        }
    }

    // If the {@link PetEntry#COLUMN_PET_GENDER} key is present,
    // check that the gender value is valid.
    if (values.containsKey(PetEntry.COLUMN_PET_GENDER)) {
        Integer gender = values.getAsInteger(PetEntry.COLUMN_PET_GENDER);
        if (gender == null || !PetEntry.isValidGender(gender)) {
            throw new IllegalArgumentException("Pet requires valid gender");
        }
    }
```

```java
    // If the {@link PetEntry#COLUMN_PET_WEIGHT} key is present,
    // check that the weight value is valid.
    if (values.containsKey(PetEntry.COLUMN_PET_WEIGHT)) {
        // Check that the weight is greater than or equal to 0 kg
        Integer weight = values.getAsInteger(PetEntry.COLUMN_PET_WEIGHT);
        if (weight != null && weight < 0) {
            throw new IllegalArgumentException("Pet requires valid weight");
        }
    }

    // No need to check the breed, any value is valid (including null).

    // If there are no values to update, then don't try to update the database
    if (values.size() == 0) {
        return 0;
    }

    // Otherwise, get writeable database to update the data
    SQLiteDatabase database = mDbHelper.getWritableDatabase();

    // Returns the number of database rows affected by the update statement
    return database.update(PetEntry.TABLE_NAME, values, selection, selectionArgs);
}
```

Run it to check if the app still run correctly.
The effect of this code will be known in next lesson

15 - Implement ContentProvider delete() method

```java
@Override
public int delete(Uri uri, String selection, String[] selectionArgs) {
    // Get writeable database
    SQLiteDatabase database = mDbHelper.getWritableDatabase();

    final int match = sUriMatcher.match(uri);
    switch (match) {
        case PETS:
            // Delete all rows that match the selection and selection args
            return database.delete(PetEntry.TABLE_NAME, selection, selectionArgs);
        case PET_ID:
            // Delete a single row given by the ID in the URI
            selection = PetEntry._ID + "=?";
            selectionArgs = new String[] { String.valueOf(ContentUris.parseId(uri)) };
            return database.delete(PetEntry.TABLE_NAME, selection, selectionArgs);
        default:
            throw new IllegalArgumentException("Deletion is not supported for " + uri);
    }
}
```

16 - Implement ContentProvider getType() method

- Step 1: Declare MIME Type constants in PetContract
- You can insert these constants right after where the pet content URI is defined in your file.
- Import android.content.ContentResolver;

```
public static final class PetEntry implements BaseColumns {

    …

    /**
     * The MIME type of the {@link #CONTENT_URI} for a list of pets.
     */
    public static final String CONTENT_LIST_TYPE =
            ContentResolver.CURSOR_DIR_BASE_TYPE + "/" + CONTENT_AUTHORITY + "/" + PATH_PETS;

    /**
     * The MIME type of the {@link #CONTENT_URI} for a single pet.
     */
    public static final String CONTENT_ITEM_TYPE =
            ContentResolver.CURSOR_ITEM_BASE_TYPE + "/" + CONTENT_AUTHORITY + "/" + PATH_PETS;
```

PetProvider.java

```
 @Override
public String getType(Uri uri) {
    final int match = sUriMatcher.match(uri);
    switch (match) {
        case PETS:
            return PetEntry.CONTENT_LIST_TYPE;
        case PET_ID:
            return PetEntry.CONTENT_ITEM_TYPE;
        default:
            throw new IllegalStateException("Unknown URI " + uri + " with match " + match);
    }
}
```

PetDbHelper.java

```
public static final String LOG_TAG = PetDbHelper.class.getSimpleName();
```