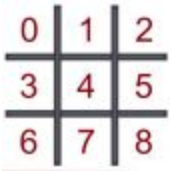
Tic-Tac-Toe

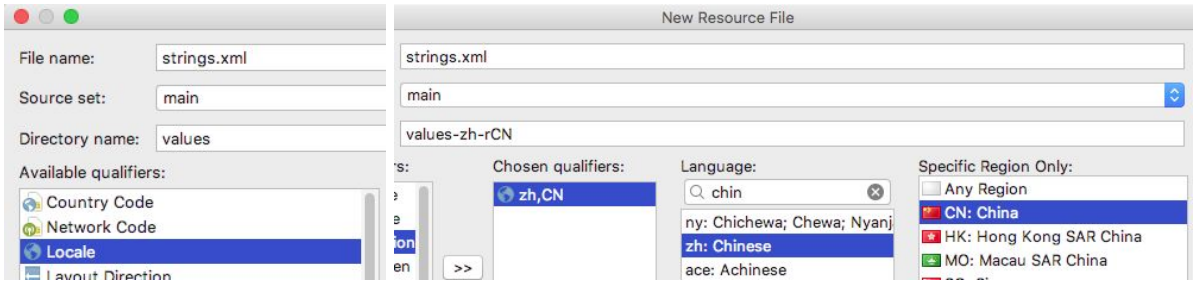
Demo a solution of the Tic-Tac-Toe game!!!!! Put the index plan on the board now:



Create a new Empty project called **Tic-Tac-Toe**
Type or copy in the strings first.

```
<resources>
  <string name="app_name">Tic-Tac-Toe</string>
  <string name="new_game">New Game</string>
  <string name="x_turn">X\'s Turn</string>
  <string name="o_turn">O\'s Turn</string>
  <string name="x_win">X Wins!</string>
  <string name="o_win">O Wins!</string>
  <string name="tie_game">Tie Game</string>
</resources>
```

Make a Chinese version again.



Make some minor changes again.

Go ahead and do the colors now too. Why not.

```
<color name="textColor">#ccc</color>
<color name="background">#800000</color>
```

Download the website

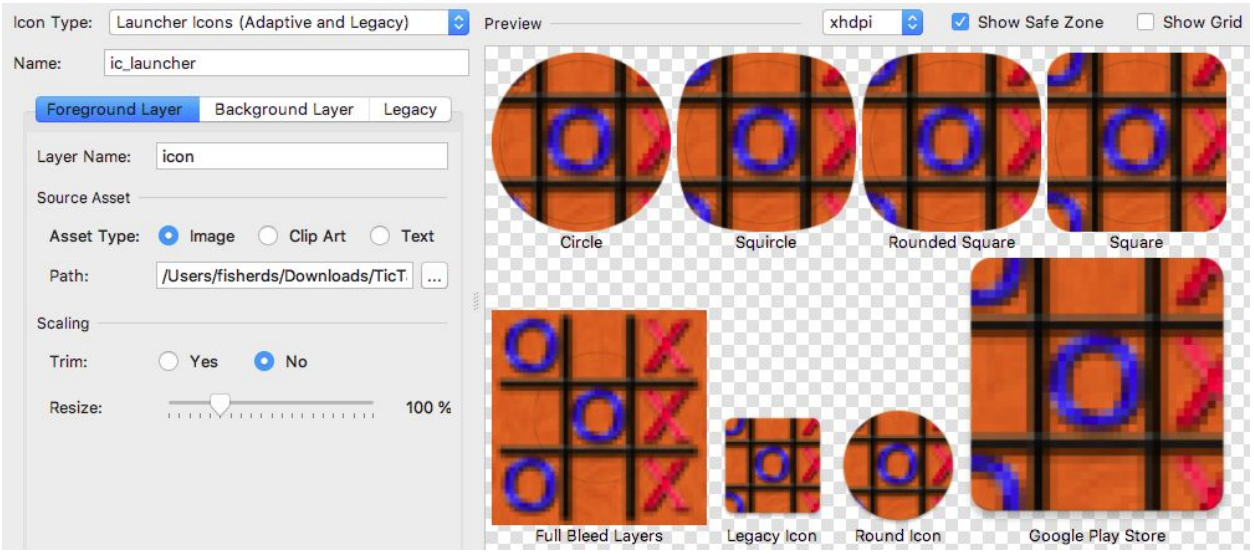
- Tic-Tac-Toe files

Copy paste the TicTacToeGame.java file into the project. Public API (**put methods on board**)

- **public** TicTacToeGame(Context context)
- **public void** pressedButtonAtIndex(**int** buttonIndex)
- **public** String stringForButtonAtIndex(**int** buttonIndex)
- **public** String stringForGameState()

Lets go ahead and all the icon file too.

Right click on the res folder, select New → Image Assets, then click "..." to find the file.



You can play with the zoom if you like, but I wouldn't bother do much. Run it. View icon.

Next make the layout (in pieces) together. This will take a bit. First make it a relative layout with a background color.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@color/background"
   tools:context=".MainActivity">
</RelativeLayout>
```

Next add a TableLayout and a single button (highlight things to notice)

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@color/background"
  tools:context=".MainActivity">

  <TableLayout
    android:id="@+id/board"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true">

    <TableRow
      android:layout_width="match_parent"
      android:layout_height="match_parent">

      <Button
        android:id="@+id/button0"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:onClick="pressedSquare"
        android:tag="0"
        android:textSize="50sp"/>
    </TableRow>
  </TableLayout>

</RelativeLayout>
```

Mass produce the buttons

activity_main.xml

```
        <Button
            android:id="@+id/button0"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:onClick="pressedSquare"
            android:tag="0"
            android:textSize="50sp"/>

        <Button
            android:id="@+id/button1"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:onClick="pressedSquare"
            android:tag="1"
            android:textSize="50sp"/>

        <Button
            android:id="@+id/button2"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:onClick="pressedSquare"
            android:tag="2"
            android:textSize="50sp"/>
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <Button
            android:id="@+id/button3"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:onClick="pressedSquare"
            android:tag="3"
            android:textSize="50sp"/>
```

…. etc up to tag 8

Finally add the game state label and New Game button

activity_main.xml

```
  <TextView
     android:id="@+id/game_state_text_view"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_above="@id/board"
     android:layout_centerHorizontal="true"
     android:layout_marginBottom="20dp"
     android:text="@string/x_turn"
     android:textColor="@color/textColor"
     android:textSize="30sp"
     android:textStyle="bold"/>

  <Button
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_alignRight="@id/board"
     android:layout_below="@id/board"
     android:layout_marginTop="20dp"
     android:onClick="pressedNewGame"
     android:text="@string/new_game"/>

</RelativeLayout>
```

Next let's connect the views to the controller.

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

 private Button[] mButtons;
 private TextView mGameStateTextView;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    mGameStateTextView = findViewById(R.id.game_state_text_view);
    mButtons = new Button[TicTacToeGame.NUM_SQUARES];
    mButtons[0] = findViewById(R.id.button0);
    mButtons[1] = findViewById(R.id.button1);
    mButtons[2] = findViewById(R.id.button2);
    mButtons[3] = findViewById(R.id.button3);
    mButtons[4] = findViewById(R.id.button4);
    mButtons[5] = findViewById(R.id.button5);
    mButtons[6] = findViewById(R.id.button6);
    mButtons[7] = findViewById(R.id.button7);
    mButtons[8] = findViewById(R.id.button8);
  }
}
```

Next make button callbacks print logs (for now)

MainActivity.java

```
private static final String TAG = "TTT";


public void pressedSquare(View view) {
 int buttonIndex = Integer.valueOf((String) view.getTag());
 Log.d(TAG, "Pressed button " + buttonIndex);
}

public void pressedNewGame(View view) {
 Log.d(TAG, "Pressed new game");
}
```

Now let's add a Game object.

MainActivity.java

```
private TicTacToeGame mGame;

protected void onCreate(Bundle savedInstanceState) {
...
 mGame = new TicTacToeGame(this);
}

public void pressedSquare(View view) {
 int buttonIndex = Integer.valueOf((String) view.getTag());
 mGame.pressedButtonAtIndex(buttonIndex);
 //Log.d(TAG, "Pressed button " + buttonIndex);
 updateView();
}


public void pressedNewGame(View view) {
 mGame = new TicTacToeGame(this);
 //Log.d(TAG, "Pressed new game");
 updateView();
}

private void updateView() {

}
```

Draw on the board the MVC diagram again and review game methods (because it'll finish in a rush).

- **public** TicTacToeGame(Context context)
- **public void** pressedButtonAtIndex(**int** buttonIndex)
- **public** String **stringForButtonAtIndex**(**int** buttonIndex)
- **public** String **stringForGameState**()

MainActivity.java

```
private void updateView() {
 mGameStateTextView.setText(mGame.stringForGameState());

 for (int i = 0; i < TicTacToeGame.NUM_SQUARES; i++) {
   mButtons[i].setText(mGame.stringForButtonAtIndex(i));
 }
}
```