Step 2: Dart classes

In this step, you create your first classes and run tests on it.

Note: Throughout this code lab, continue to edit the files in `s1_basics`. You can use the files in the other `samples` subdirectories to compare to your code or to recover if you get off track.

***Keywords**: class, test*

Create `Country` class

Edit `lib/src/map.dart`, as follows.

→ Add the following class:

```
/// Country class.
class Country {
  /// The country id.
  final String id;
  /// The country neighbours.
  final List<String> neighbours;

  Country(this.id, this.neighbours);
}
```

You've just implemented a class!

Key information:

- It defines `Country` class with two fields: `id` which is the country Id and `neighbours` which is the list of country neighbours.
- The fields are `final`, it means that they can only be initialized once, in constructors.
- Generic types allow to define a `List` of `String` with `List<String>`.
- The constructor `Country(this.id, this.neighbours);` use syntactic sugar for setting `id` and `neighbours` before the constructor body runs.
- Lines starting with `///` or `/**` are documentation comments, used to generate documentation.

Instantiate a `Country` class

Edit `bin/main.dart`, as follows.

→ Instantiate a `Country` class and run this code:

```
library risk.main;

import '../lib/risk.dart';

main() {
  Country country = new Country('eastern_australia', ['western_australia', 'new_guinea']);
  var neighbours = country.neighbours;
  print("Hello ${country.id} and $neighbours!");
}
```

- The `import` is used to import a namespace from one library in the local scope.
- The `../lib/risk.dart` import is a library defined in the `lib` directory. `lib/src/map.dart` is already part of this library.
- Either `'string'` or `"string"` can be used to define a `String`.
- `['western_australia', 'new_guinea']` allows to easily define `List`.
- `var` is a way to declare a variable without specifying its type.
- `$neighbours` and `${country.id}` are string interpolations. It includes the variable or expression's string equivalent inside of a string literal.

Create `Continent` class

Continue to edit `lib/src/map.dart`.

→ Implements the `Continent`:

- The `Continent` class has three fields `id`, `bonus` and `countries`:
  - `id` (a `String`) is the continent id.
  - `bonus` (an `int`) is the reinforcement bonus given if the same player owns all continent countries.
  - `countries` (a `List` of `String`) is the list of the ids of the countries of this continent.
- Like `Country`, fields must be immutable.
- The `Continent` class must be instanciable with the following instruction:

```
new Continent('australia', 2, ["eastern_australia", "indonesia"])
```

Add `Country` and `Continent` constants

Edit `lib/src/map.dart`, as follows.

→ Add the following top-level constants:

```
/// List of all existing countries
final List<Country> COUNTRIES = [//
  // australia
  new Country('eastern_australia', ['western_australia', 'new_guinea', 'eastern_australia']), //
  new Country('indonesia', ['siam', 'new_guinea', 'western_australia']), //
  new Country('new_guinea', ['indonesia', 'eastern_australia', 'western_australia']), //
  new Country('western_australia', ['eastern_australia', 'indonesia', 'new_guinea']), //
  // south_america
  new Country('argentina', ['brazil', 'peru']), //
  new Country('brazil', ['north_africa', 'venezuela', 'argentina', 'peru']), //
  new Country('peru', ['venezuela', 'brazil', 'argentina']), //
  new Country('venezuela', ['central_america', 'brazil', 'peru']), //
  // africa
  new Country('congo', ['east_africa', 'south_africa', 'north_africa']), //
  new Country('egypt', ['southern_europe', 'middle_east', 'east_africa', 'north_africa']), //
  new Country('east_africa', ['middle_east', 'madagascar', 'south_africa', 'congo', 'north_africa', 'egyp
  new Country('madagascar', ['east_africa', 'south_africa']), //
  new Country('north_africa', ['southern_europe', 'western_europe', 'brazil', 'egypt', 'east_africa', 'co
  new Country('south_africa', ['madagascar', 'east_africa', 'congo']), //
  // north_america
  new Country('alaska', ['kamchatka', 'northwest_territory', 'alberta']), //
  new Country('alberta', ['alaska', 'ontario', 'northwest_territory', 'western_united_states']), //
  new Country('central_america', ['venezuela', 'eastern_united_states', 'western_united_states']), //
  new Country('eastern_united_states', ['quebec', 'ontario', 'western_united_states', 'central_america'])
  new Country('greenland', ['iceland', 'ontario', 'northwest_territory', 'quebec']), //
  new Country('northwest_territory', ['alaska', 'ontario', 'greenland', 'alberta']), //
  new Country('ontario', ['northwest_territory', 'greenland', 'eastern_united_states', 'western_united_st
  new Country('quebec', ['greenland', 'ontario', 'eastern_united_states']), //
  new Country('western_united_states', ['alberta', 'ontario', 'eastern_united_states', 'central_america']
  // europe
  new Country('great_britain', ['iceland', 'western_europe', 'northern_europe', 'scandinavia']), //
  new Country('iceland', ['greenland', 'great_britain', 'scandinavia']), //
  new Country('northern_europe', ['ukraine', 'scandinavia', 'great_britain', 'western_europe', 'southern_
  new Country('scandinavia', ['iceland', 'great_britain', 'northern_europe', 'ukraine']), //
  new Country('southern_europe', ['north_africa', 'egypt', 'middle_east', 'ukraine', 'northern_europe', '
  new Country('ukraine', ['ural', 'afghanistan', 'middle_east', 'southern_europe', 'northern_europe', 'sc
  new Country('western_europe', ['north_africa', 'southern_europe', 'northern_europe', 'great_britain']),
  // asia
  new Country('afghanistan', ['ukraine', 'ural', 'china', 'india', 'middle_east']), //
  new Country('china', ['siberia', 'ural', 'afghanistan', 'india', 'siam', 'mongolia']), //
  new Country('india', ['middle_east', 'afghanistan', 'china', 'siam']), //
  new Country('irkutsk', ['siberia', 'kamchatka', 'mongolia', 'yakursk']), //
  new Country('japan', ['kamchatka', 'mongolia']), //
  new Country('kamchatka', ['alaska', 'yakursk', 'irkutsk', 'mongolia', 'japan']), //
  new Country('ural', ['siberia', 'china', 'afghanistan', 'ukraine']), //
  new Country('middle_east', ['southern_europe', 'ukraine', 'afghanistan', 'india', 'egypt', 'east_africa
  new Country('mongolia', ['china', 'siberia', 'irkutsk', 'kamchatka', 'japan']), //
  new Country('siam', ['india', 'china', 'indonesia']), //
  new Country('siberia', ['yakursk', 'irkutsk', 'mongolia', 'china', 'ural']), //
  new Country('yakursk', ['kamchatka', 'siberia', 'irkutsk', 'eastern_australia']),//
  new Country('indonesia', ['siam', 'new_guinea', 'western_australia']), //
];

/// All [Country]s indexed by country id
final Map<String, Country> COUNTRY_BY_ID = new Map.fromIterable(COUNTRIES, key: (country) => country.id);

/// List of all existing continents
final List<Continent> CONTINENTS = [//
  new Continent('australia', 2, ["eastern_australia", "indonesia", "new_guinea"]), //
  new Continent('north_america', 5, ["alaska", "alberta", "central_america", "eastern_united_states", "gr
      "western_united_states"]), //
  new Continent('south_america', 2, ["argentina", "brazil", "peru", "venezuela"]), //
  new Continent('africa', 3, ["congo", "egypt", "east_africa", "madagascar", "north_africa", "south_afric
  new Continent('europe', 5, ["great_britain", "iceland", "northern_europe", "scandinavia", "southern_eur
  new Continent('asia', 7, ["afghanistan", "china", "india", "irkutsk", "japan", "kamchatka", "ural", "mi
];
```

Key information:

- Those constants defines the countries and continents for the Risk game.

- `COUNTRY_BY_ID` is a Map built from `COUNTRIES`. It helps to find quickly a `Country` by its `id`.
- **Some errors are hidden in those constants, they need to be fixed!**

Run tests on the implementations

To check if `Country` and `Continent` classes are well implemented and to fix errors in constants, it's a good pratice to run unit tests.

Open `test/s2_classes_test.dart`.

→ Get familiar with the code:

```
library risk.map.test;

import 'package:unittest/unittest.dart';
import '../lib/risk.dart';

main() {
  test('Country should be instanciable', () {
    var country = new Country('eastern_australia', ['western_australia', 'new_guinea']);
    expect(country, isNotNull);
    expect(country.id, equals('eastern_australia'));
    expect(country.neighbours, equals(['western_australia', 'new_guinea']));
  });

  group('COUNTRIES', () {
    test('for each country should have at least 1 neighbour', () { ... });
    // ...
  });

  group('COUNTRY_BY_ID', () { ... });
  group('CONTINENTS', () { ... });
}
```

→ Then run tests: right-click `test/s2_classes_test.dart` and select **Run**.
→ **You should have test failures in console, fix classes and/or constants**.

Key information:

- Note how a test is written: we wrap it in a call to `test(String testName, functionToTest);`.
- Within the function we are testing, we write assertions, using `expect(actualValue, expectedValueMatcher);`.
- The `unittest` package provides a set of built-in `Matcher` like `equals(xxx)`, `isPositive`, `isNotNull`, `isTrue` ...
- It can be helpful to group similar tests together, which can be done with `group()`.
- Use the `solo_` prefix to quickly run just one unit test or one group: `solo_test(...)` or `solo_group(...)`
- Use the `skip_` prefix to exclude unit tests or groups you don't want to run: `skip_test(...)` or `skip_group(...)`

Write a test

Open `test/s2_classes_test.dart`.

→ Test that `COUNTRIES` and `COUNTRY_BY_ID` have exactly 42 countries.
→ Fix `COUNTRIES` removing the duplicated country.

Learn more

- Dart Language - Classes
- Dart Language - Documentation Comments
- Unit Testing with Dart

Problems?

Check your code against the files in s2_classes (diff).

Home | < Previous | Next >