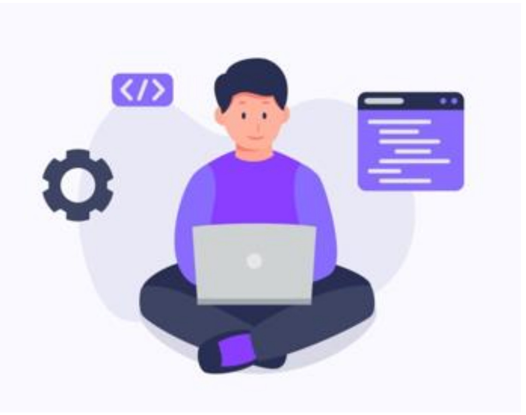


Session 7
Packages in Dart

Session Overview

- Define Dart package
- Describe how packages are used
- Explain types of Dart packages
- Explain creation of a Dart package

Dart Package

- Dart package is a set of files that forms a software library.
- Some of built-in packages in Dart are `math`, `html`, and `collections`.
- Developers can create user-defined packages as well.
- Dart packages helps to solve problems without writing code from scratch.

pub Commands

Command	Description
`pub get`	Utilized to get all the packages that the application depends on.
`pub upgrade`	Utilized to upgrade all the packages that the application depends on.
`pub help`	Utilized to get all the pub commands that are available.

Types of Dart Packages

- General-Purpose packages
- Dart Core Library packages
- Specialized packages

Plug-in Packages

- These are specific packages that have API and written using Dart code.
- Plug-in packages are written for operating systems and Web applications.
- As an example a built-in Dart package `dart:io` is used for reading and writing a file.

Code Snippet 1:

```
import 'dart:io ';
//To create a file from a URL
File newFile =
File.fromUri(Uri.parse('file url from Web'));
```

Creation of User-Defined Packages

Users can create their own packages to meet their requirements.

The directory and structure for a package is automatically created using `create` command.

Syntax for creating package at the terminal:

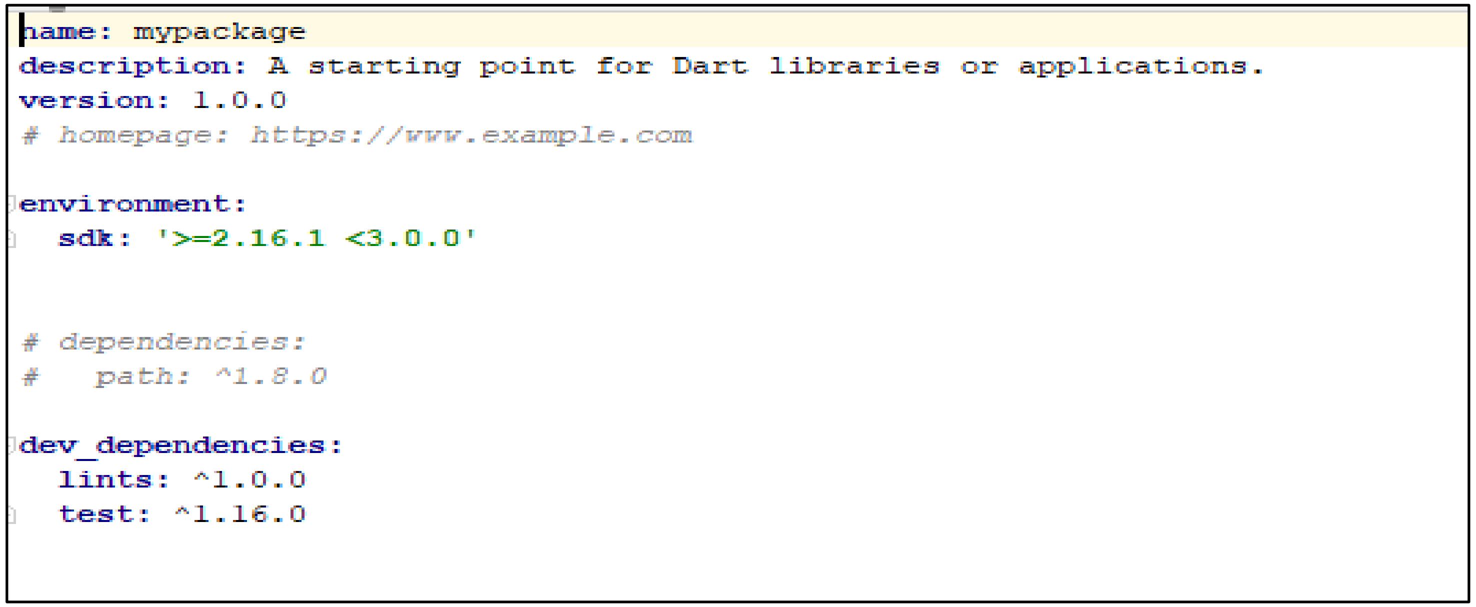
```
dart create --template=package-simple  mypackage
```

Syntax for importing package inside a project:

```
'package:mypackage/main.dart’;
```

pubspec.yaml File

This is the structure of `pubspec.yaml` file, that is created during package creation:



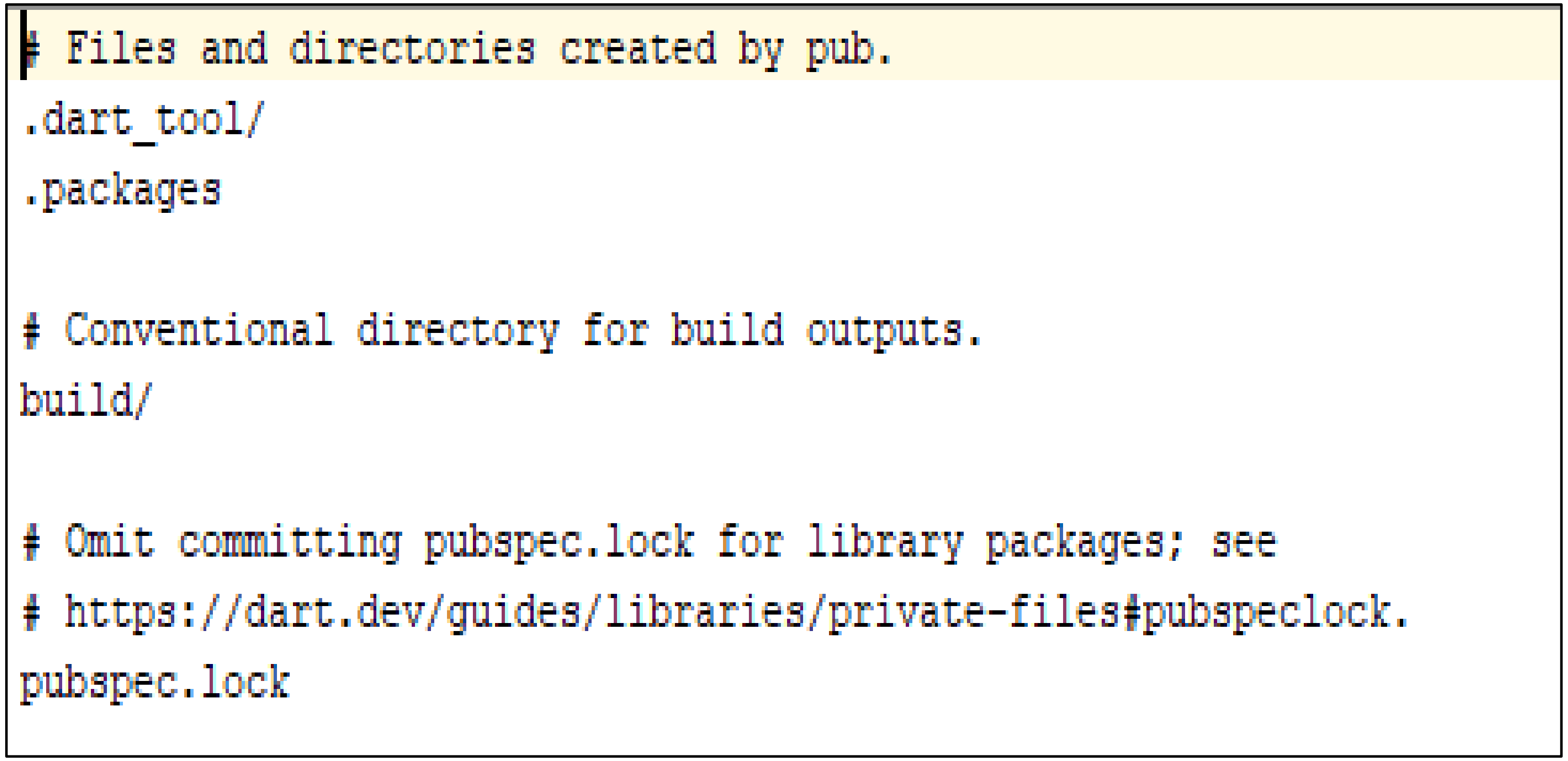
```
name: mypackage
description: A starting point for Dart libraries or applications.
version: 1.0.0
# homepage: https://www.example.com

environment:
  sdk: '>=2.16.1 <3.0.0'


# dependencies:
#   path: ^1.8.0

dev_dependencies:
  lints: ^1.0.0
  test: ^1.16.0
```

Code Snippet 2:

```
import 'package:mypackage/main.dart';
```

.gitignore File

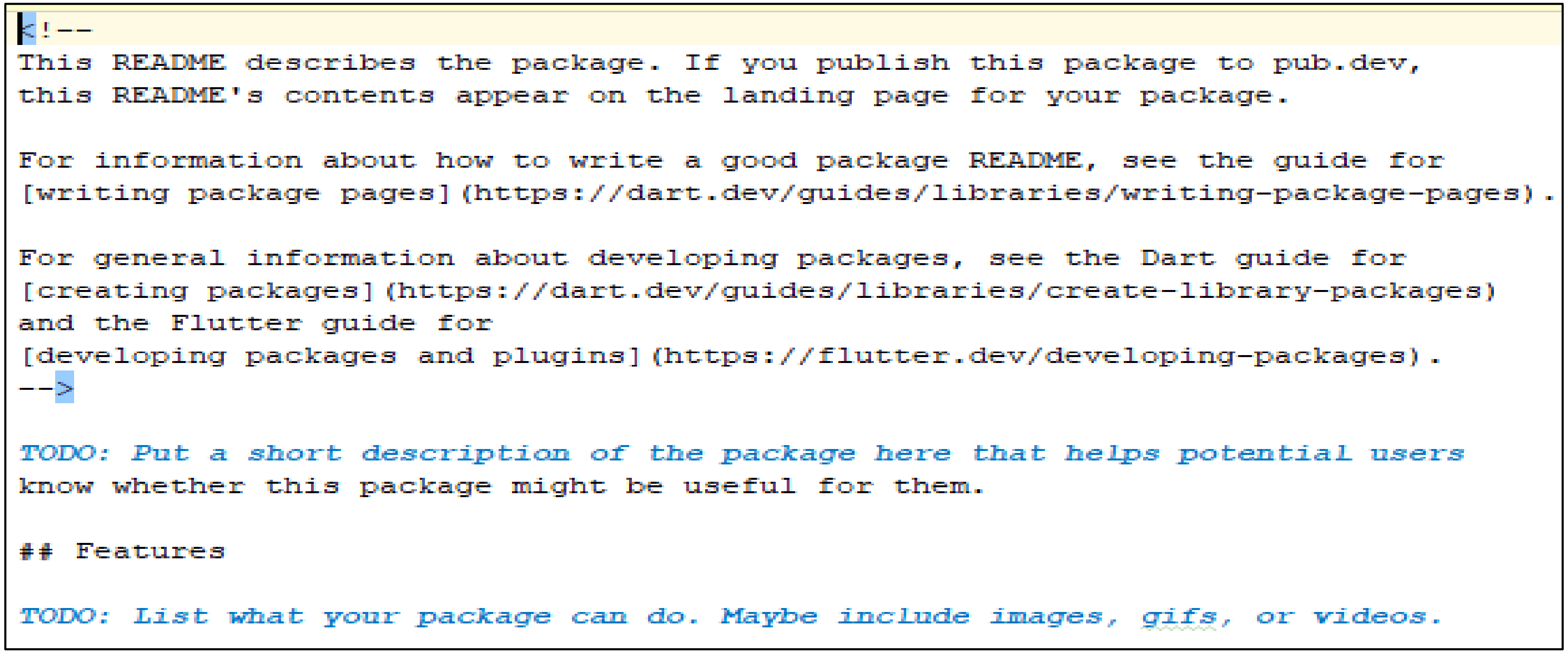
This is the structure of .gitignore file, that is created during package creation:



```
# Files and directories created by pub.
.dart_tool/
.packages

# Conventional directory for build outputs.
build/

# Omit committing pubspec.lock for library packages; see
# https://dart.dev/guides/libraries/private-files#pubspeclock.
pubspec.lock
```

README.md File

Following file is the README.md file, that is created during package creation:



```
<!--
This README describes the package. If you publish this package to pub.dev,
this README's contents appear on the landing page for your package.

For information about how to write a good package README, see the guide for
[writing package pages](https://dart.dev/guides/libraries/writing-package-pages).

For general information about developing packages, see the Dart guide for
[creating packages](https://dart.dev/guides/libraries/create-library-packages)
and the Flutter guide for
[developing packages and plugins](https://flutter.dev/developing-packages).
-->

TODO: Put a short description of the package here that helps potential users
know whether this package might be useful for them.

## Features

TODO: List what your package can do. Maybe include images, gifs, or videos.
```

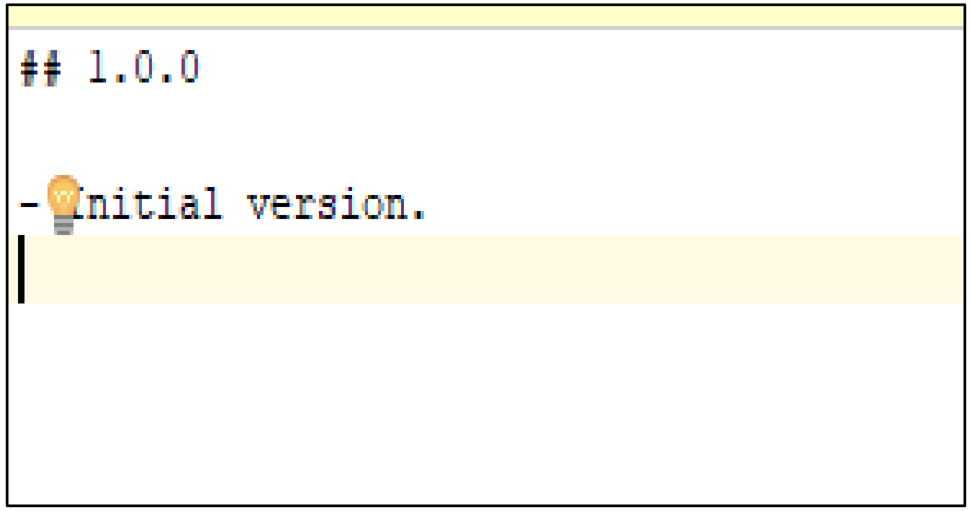
CHANGELOG.md File

This file give latest updates that are done on the project.

The changes are organized according to the date and version.

Improvements or bugs can be identified in the project using this file.

Following file is the CHANGELOG.md file, that is created during package creation:



```
## 1.0.0

- Initial version.
```

Usage of User-Defined Package [1-2]

A class called Arithmetic is created inside the package mypackage.

This class is created inside `lib/src/mypackage_base.dart`

Code Snippet 3:

```
class Arithmetic {
static int add(int a, int b) => a + b;
static int sub(int a, int b) => a + b;
static int multiply(int a, int b) => a + b;
static double divide(int a, int b) => a / b;
}
```

Usage of User-Defined Package [2-2]

Command for creating new package:

```
dart create sample_dart_project
```

Code Snippet 4:

```
dependencies:
  mypackage:
    path: /Users/rax/Dart/mypackage
```

Code Snippet 5:

```
import 'package:mypackage/mypackage.dart';

void main() {
  int a = 10;
  int b = 6;
  print(Arithmetic.add(a, b));
  print(Arithmetic.sub(a, b));
  print(Arithmetic.multiply(a, b));
  print(Arithmetic.divide(a, b));
}
```

Summary

- Dart package is an extremely coherent, reusable, and unconstrained code unit.
- Dart has a set of default packages that get loaded automatically during the starting of the Dart console.
- Packages are generally reusable codes that can be used for many applications across all the platforms.
- Packages are utilized by Dart to share libraries and tools.
- General purpose packages such as `http` and `intl` are used in a wide range of projects.
- Packages expanding on core Dart libraries such as `io`, `collection`, and `async` are built on the Dart core library.
- Plugin packages are specific packages that have an API that is written using Dart code along with one or more than one implementations that are platform specific.