



Avro

---

# Avro

Apache Avro Data  
Serialization

---

# Apache Avro



- ❖ Data serialization system
- ❖ Data structures
- ❖ Binary data format
- ❖ Container file format to store persistent data
- ❖ RPC capabilities
- ❖ Does not require code generation to use

# Avro Schemas



- ❖ Supports schemas for defining data structure
- ❖ Serializing and deserializing data, uses schema
- ❖ File schema
  - ❖ Avro files store data with its schema
- ❖ RPC Schema
  - ❖ RPC protocol exchanges schemas as part of the handshake
- ❖ Schemas written in JSON

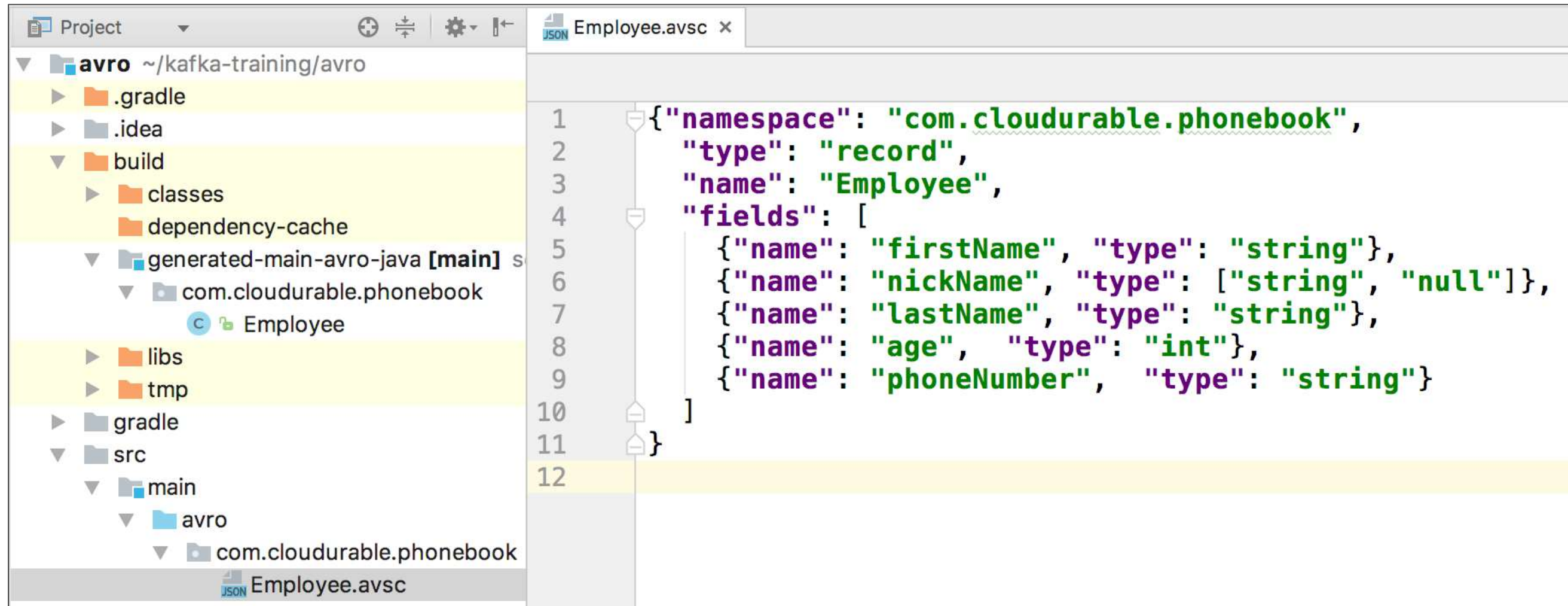
# Avro compared to...

---



- ❖ Similar to Thrift, Protocol Buffers, JSON, etc.
- ❖ Does not require code generation
- ❖ Avro needs less encoding as part of the data since it stores names and types in the schema
- ❖ It supports evolution of schemas.

# Avro Schema

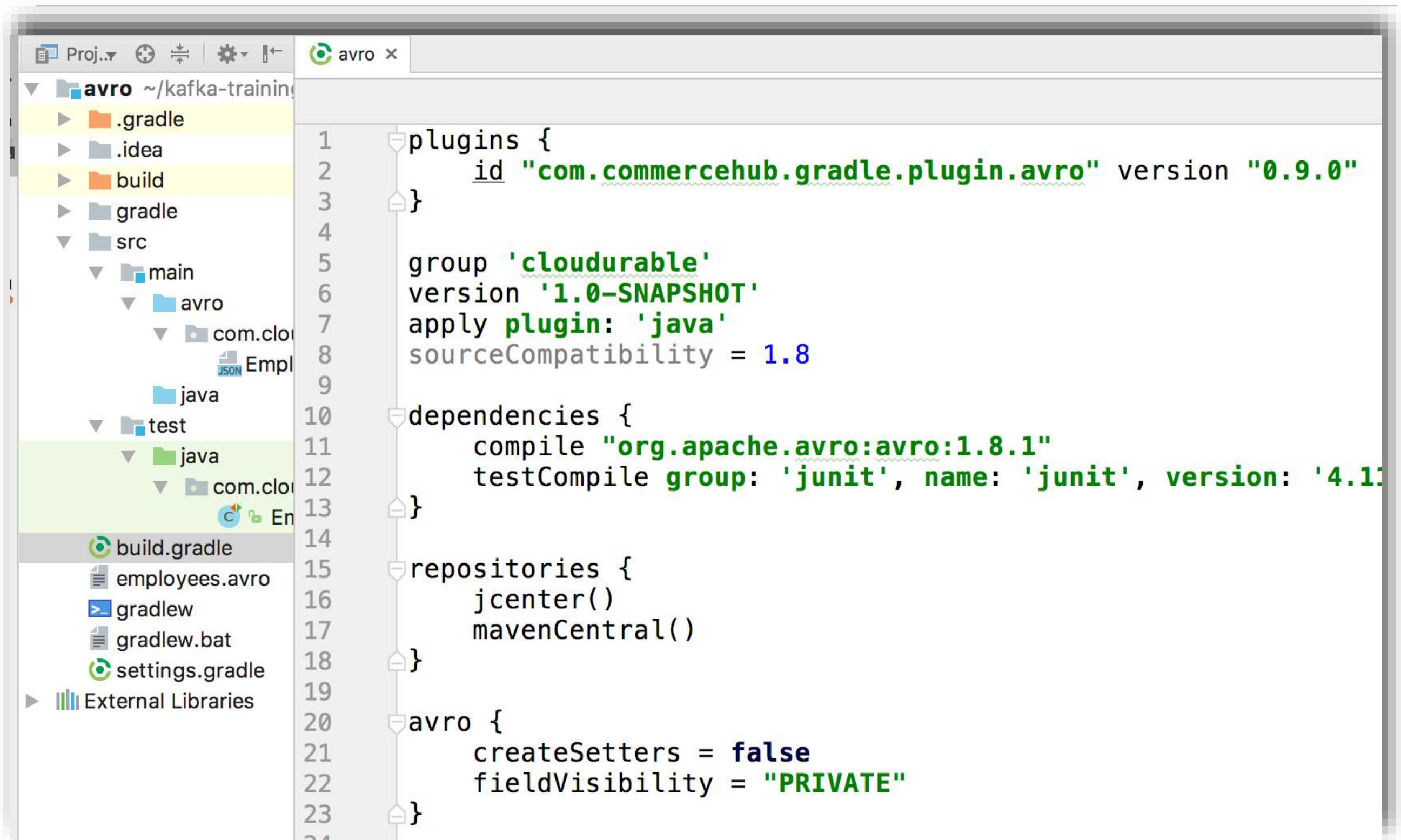


The screenshot shows an IDE window with a project explorer on the left and a code editor on the right. The project explorer displays the directory structure of a project named 'avro' located at '~/kafka-training/avro'. The 'src/main/avro' directory is expanded, showing a package 'com.cloudurable.phonebook' which contains a file 'Employee'. The code editor displays the 'Employee.avsc' file, which contains the following JSON schema:

```
1  {"namespace": "com.cloudurable.phonebook",
2    "type": "record",
3    "name": "Employee",
4    "fields": [
5      {"name": "firstName", "type": "string"},
6      {"name": "nickName", "type": ["string", "null"]},
7      {"name": "lastName", "type": "string"},
8      {"name": "age", "type": "int"},
9      {"name": "phoneNumber", "type": "string"}
10   ]
11 }
12
```

Avro schema stored in src/main/avro by default.

# Code Generation

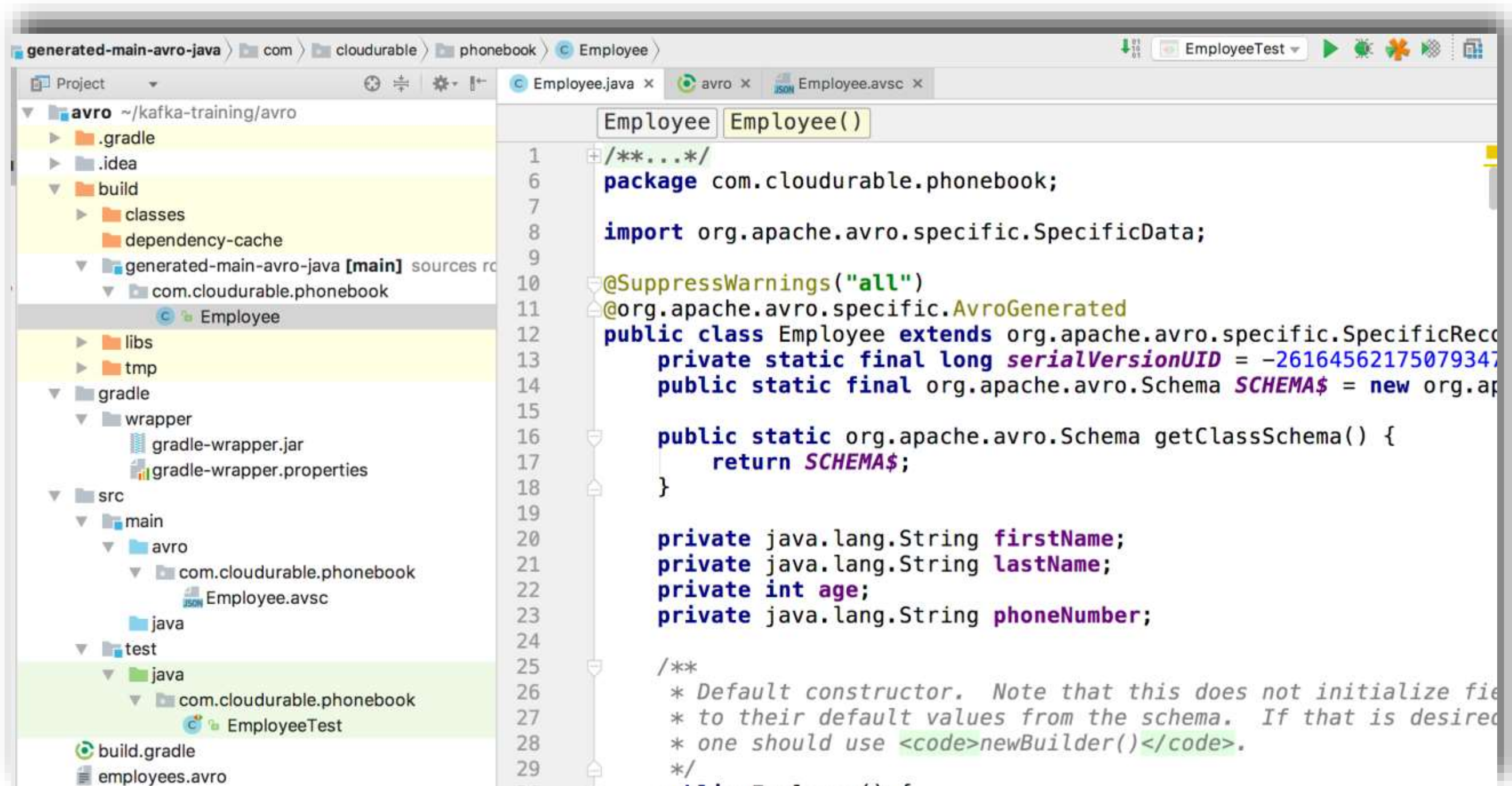
The screenshot shows an IDE window with a project named 'avro' located at '~/kafka-training'. The file explorer on the left shows the project structure, including folders for '.gradle', '.idea', 'build', 'gradle', 'src', 'main', 'test', and 'java'. The 'build.gradle' file is selected and open in the editor. The code in the editor is a Gradle build script for an Avro project.

```

1  plugins {
2      id "com.commercehub.gradle.plugin.avro" version "0.9.0"
3  }
4
5  group 'cloudurable'
6  version '1.0-SNAPSHOT'
7  apply plugin: 'java'
8  sourceCompatibility = 1.8
9
10 dependencies {
11     compile "org.apache.avro:avro:1.8.1"
12     testCompile group: 'junit', name: 'junit', version: '4.11'
13 }
14
15 repositories {
16     jcenter()
17     mavenCentral()
18 }
19
20 avro {
21     createSetters = false
22     fieldVisibility = "PRIVATE"
23 }
  
```



# Employee Code Generation

The screenshot shows an IDE with the following components:

- Project Explorer (Left):** Displays the project structure for 'generated-main-avro-java'. The 'src/main/avro' directory is expanded, showing the 'com.cloudurable.phonebook' package containing the 'Employee.avsc' schema file.
- Editor (Right):** Shows the generated Java code for the 'Employee' class. The code is as follows:
 

```

1  /**...*/
6  package com.cloudurable.phonebook;
7
8  import org.apache.avro.specific.SpecificData;
9
10 @SuppressWarnings("all")
11 @org.apache.avro.specific.AvroGenerated
12 public class Employee extends org.apache.avro.specific.SpecificRecord {
13     private static final long serialVersionUID = -26164562175079347;
14     public static final org.apache.avro.Schema SCHEMA$ = new org.ap
15
16     public static org.apache.avro.Schema getClassSchema() {
17         return SCHEMA$;
18     }
19
20     private java.lang.String firstName;
21     private java.lang.String lastName;
22     private int age;
23     private java.lang.String phoneNumber;
24
25     /**
26      * Default constructor. Note that this does not initialize fields
27      * to their default values from the schema. If that is desired
28      * one should use <code>newBuilder()</code>.
29      */

```

# Using Generated Avro class



```
Employee bob = Employee.newBuilder().setAge(35)
    .setFirstName("Bob")
    .setLastName("Jones")
    .setPhoneNumber("555-555-1212")
    .build();

assertEquals( expected: "Bob", bob.getFirstName());
```





# Writing employees to an Avro File

```
final DatumWriter<Employee> datumWriter = new SpecificDatumWriter<>(Employee.class);
final DataFileWriter<Employee> dataFileWriter = new DataFileWriter<>(datumWriter);

try {
    dataFileWriter.create(employeeList.get(0).getSchema(),
        new File( pathname: "employees.avro" ));
    employeeList.forEach(employee -> {
        try {
            dataFileWriter.append(employee);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });
} finally {
    dataFileWriter.close();
}
```

# Reading employees From a File



```
final File file = new File( pathname: "employees.avro");
final List<Employee> employeeList = new ArrayList<>();
final DatumReader<Employee> empReader = new SpecificDatumReader<>(Employee.class);
final DataFileReader<Employee> dataFileReader = new DataFileReader<>(file, empReader);

while (dataFileReader.hasNext()) {
    employeeList.add(dataFileReader.next(new Employee()));
}

employeeList.forEach(System.out::println);
```

All 2 tests passed – 333ms

/Library/Java/JavaVirtualMachines/jdk1.8.0\_66.jdk/Contents/Home/bin/java ...

```
{"firstName": "Bob0", "lastName": "Jones0", "age": 25, "phoneNumber": "555-555-1212"}
{"firstName": "Bob1", "lastName": "Jones1", "age": 26, "phoneNumber": "555-555-1212"}
{"firstName": "Bob2", "lastName": "Jones2", "age": 27, "phoneNumber": "555-555-1212"}
{"firstName": "Bob3", "lastName": "Jones3", "age": 28, "phoneNumber": "555-555-1212"}
{"firstName": "Bob4", "lastName": "Jones4", "age": 29, "phoneNumber": "555-555-1212"}
{"firstName": "Bob5", "lastName": "Jones5", "age": 30, "phoneNumber": "555-555-1212"}
{"firstName": "Bob6", "lastName": "Jones6", "age": 31, "phoneNumber": "555-555-1212"}
{"firstName": "Bob7", "lastName": "Jones7", "age": 32, "phoneNumber": "555-555-1212"}
{"firstName": "Bob8", "lastName": "Jones8", "age": 33, "phoneNumber": "555-555-1212"}
{"firstName": "Bob9", "lastName": "Jones9", "age": 34, "phoneNumber": "555-555-1212"}
{"firstName": "Bob10", "lastName": "Jones10", "age": 35, "phoneNumber": "555-555-1212"}
{"firstName": "Bob11", "lastName": "Jones11", "age": 36, "phoneNumber": "555-555-1212"}
{"firstName": "Bob12", "lastName": "Jones12", "age": 37, "phoneNumber": "555-555-1212"}
```

# Using GenericRecord



```
final String schemaLoc = "src/main/avro/com/cloudurable/phonebook/Employee.avsc";
final File schemaFile = new File(schemaLoc);
final Schema schema = new Schema.Parser().parse(schemaFile);

GenericRecord bob = new GenericData.Record(schema);
bob.put(key: "firstName", v: "Bob");
bob.put(key: "lastName", v: "Smith");
bob.put(key: "phoneNumber", v: "555-555-1212");
bob.put(key: "age", v: 35);

assertEquals(expected: "Bob", bob.get("firstName"));
```

# Writing Generic Records



```
final List<GenericRecord> employeeList = new ArrayList<>();
```

```
final DatumWriter<GenericRecord> datumWriter = new GenericDatumWriter<>(schema);  
final DataFileWriter<GenericRecord> dataFileWriter = new DataFileWriter<>(datumWriter);  
try {  
    dataFileWriter.create(employeeList.get(0).getSchema(),  
        new File( pathname: "employees2.avro"));  
    employeeList.forEach(employee -> {  
        try {  
            dataFileWriter.append(employee);  
        } catch (IOException e) {  
            throw new RuntimeException(e);  
        }  
    });  
} finally {  
    dataFileWriter.close();  
}
```



# Reading using Generic Records



```
final File file = new File( pathname: "employees2.avro");
final List<GenericRecord> employeeList = new ArrayList<>();
final DatumReader<GenericRecord> empReader = new GenericDatumReader<>();
final DataFileReader<GenericRecord> dataFileReader = new DataFileReader<>(file, empReader);

while (dataFileReader.hasNext()) {
    employeeList.add(dataFileReader.next( reuse: null));
}

employeeList.forEach(System.out::println);
```

All 2 tests passed – 307ms

/Library/Java/JavaVirtualMachines/jdk1.8.0\_66.jdk/Contents/Home/bin/java ...

```
{"firstName": "Bob0", "nickName": null, "lastName": "Smith0", "age": 25, "phoneNumber": "555-555-1212"}
{"firstName": "Bob1", "nickName": null, "lastName": "Smith1", "age": 26, "phoneNumber": "555-555-1212"}
{"firstName": "Bob2", "nickName": null, "lastName": "Smith2", "age": 27, "phoneNumber": "555-555-1212"}
{"firstName": "Bob3", "nickName": null, "lastName": "Smith3", "age": 28, "phoneNumber": "555-555-1212"}
{"firstName": "Bob4", "nickName": null, "lastName": "Smith4", "age": 29, "phoneNumber": "555-555-1212"}
{"firstName": "Bob5", "nickName": null, "lastName": "Smith5", "age": 30, "phoneNumber": "555-555-1212"}
{"firstName": "Bob6", "nickName": null, "lastName": "Smith6", "age": 31, "phoneNumber": "555-555-1212"}
{"firstName": "Bob7", "nickName": null, "lastName": "Smith7", "age": 32, "phoneNumber": "555-555-1212"}
{"firstName": "Bob8", "nickName": null, "lastName": "Smith8", "age": 33, "phoneNumber": "555-555-1212"}
{"firstName": "Bob9", "nickName": null, "lastName": "Smith9", "age": 34, "phoneNumber": "555-555-1212"}
```



# Avro Schema Validation



```
GenericRecord employee = new GenericData.Record(schema);
employee.put( key: "firstName", v: "Bob" + index);
employee.put( key: "lastName", v: "Smith" + index);
employee.put( key: "phoneNumber", v: "555-555-1212");
//employee.put("age", index % 35 + 25);
employee.put( key: "age", v: "old");
employeeList.add(employee);
```

2 tests done: 1 failed – 395ms

org.apache.avro.file.DataFileWriter\$AppendWriteException: java.lang.ClassCastException: java.lang.String cannot

at org.apache.avro.file.DataFileWriter.append(DataFileWriter.java:308)

at com.cloudurable.phonebook.EmployeeTestNoGen.lambda\$testWrite\$0(EmployeeTestNoGen.java:71)

at java.util.ArrayList.forEach(ArrayList.java:1249)

+ at com.cloudurable.phonebook.EmployeeTestNoGen.testWrite(EmployeeTestNoGen.java:69) <27 internal calls>

Caused by: java.lang.ClassCastException: java.lang.String cannot be cast to java.lang.Number

at org.apache.avro.generic.GenericDatumWriter.writeWithoutConversion(GenericDatumWriter.java:117)

at org.apache.avro.generic.GenericDatumWriter.write(GenericDatumWriter.java:73)

at org.apache.avro.generic.GenericDatumWriter.writeField(GenericDatumWriter.java:153)

at org.apache.avro.generic.GenericDatumWriter.writeRecord(GenericDatumWriter.java:143)

at org.apache.avro.generic.GenericDatumWriter.writeWithoutConversion(GenericDatumWriter.java:105)

# Avro supported types

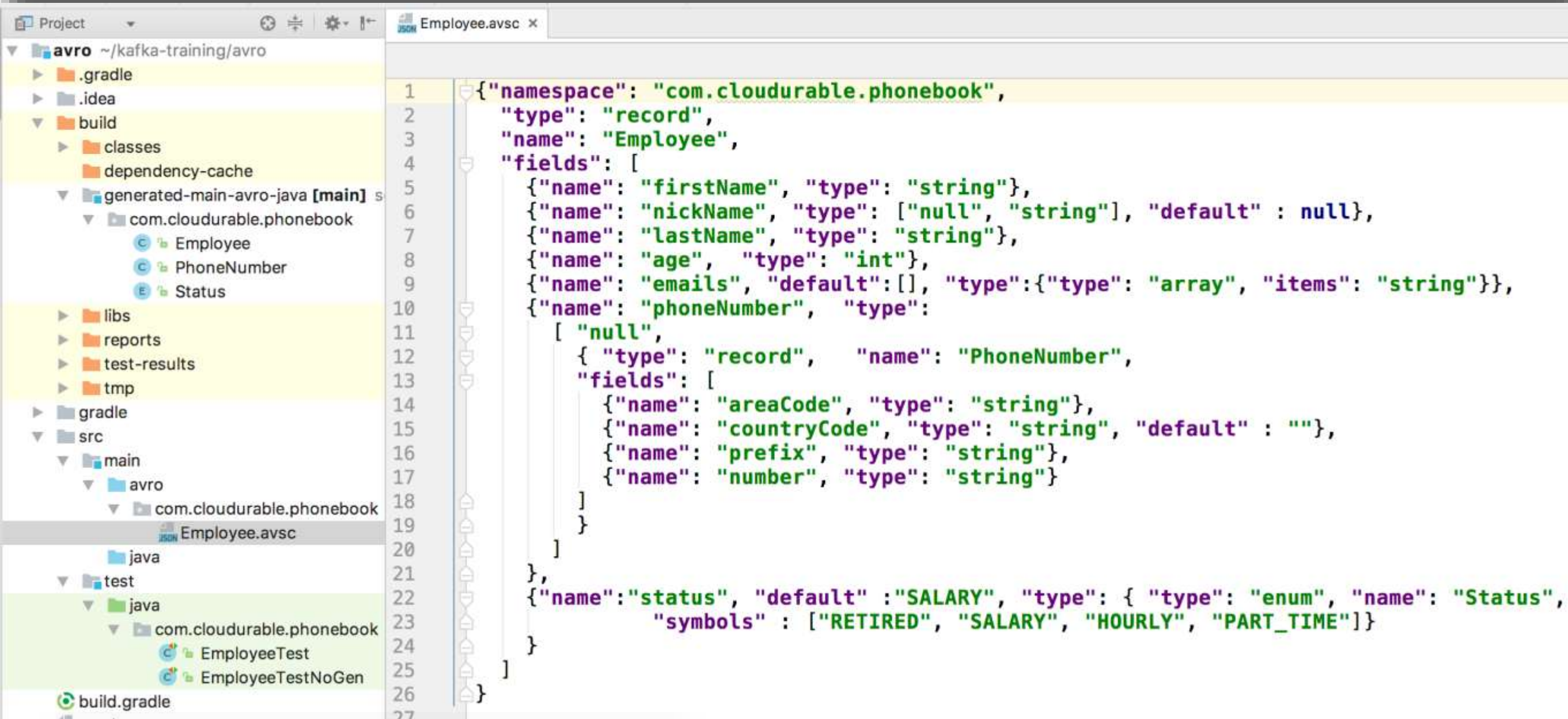
---



- ❖ Records
- ❖ Arrays
- ❖ Enums
- ❖ Unions
- ❖ Maps
- ❖ Strings, Int, Boolean, Decimal, Timestamp, Date



# Fuller example Avro Schema



The screenshot shows an IDE with a project structure on the left and an Avro schema file named `Employee.avsc` open in the editor. The project structure includes a `build` directory, `generated-main-avro-java` (containing `com.cloudurable.phonebook` with `Employee`, `PhoneNumber`, and `Status` classes), `libs`, `reports`, `test-results`, `tmp`, `gradle`, `src` (with `main` and `test` subdirectories), and `build.gradle`.

The `Employee.avsc` schema is defined as follows:

```

1  {"namespace": "com.cloudurable.phonebook",
2   "type": "record",
3   "name": "Employee",
4   "fields": [
5     {"name": "firstName", "type": "string"},
6     {"name": "nickName", "type": ["null", "string"], "default" : null},
7     {"name": "lastName", "type": "string"},
8     {"name": "age", "type": "int"},
9     {"name": "emails", "default": [], "type": {"type": "array", "items": "string"}},
10    {"name": "phoneNumber", "type":
11      [ "null",
12        { "type": "record", "name": "PhoneNumber",
13          "fields": [
14            {"name": "areaCode", "type": "string"},
15            {"name": "countryCode", "type": "string", "default" : ""},
16            {"name": "prefix", "type": "string"},
17            {"name": "number", "type": "string"}
18          ]
19        }
20      ]
21    },
22    {"name": "status", "default" : "SALARY", "type": { "type": "enum", "name": "Status",
23      "symbols" : ["RETIRED", "SALARY", "HOURLY", "PART_TIME"]}
24  ]
25  }
26  ]
27  }
```

```

public class PhoneNumber extends org.apache.avro.specific.SpecificRecord {
    private static final long serialVersionUID = -313877;
    public static final org.apache.avro.Schema SCHEMA$ =
    public static org.apache.avro.Schema getClassSchema() {
        private java.lang.String areaCode;
        private java.lang.String countryCode;
        private java.lang.String prefix;
        private java.lang.String number;
    }
}
```

```

package com.cloudurable.phonebook;
@SuppressWarnings("all")
@org.apache.avro.specific.AvroGenerated
public enum Status {
    RETIRED, SALARY, HOURLY, PART_TIME ;
    public static final org.apache.avro.Schema SCHEMA$ =
}
```

# Avro



- ❖ Fast data serialization
- ❖ Supports data structures
- ❖ Supports Records, Maps, Array, and basic types
- ❖ You can use it direct or use Code Generation
- ❖ [Read more](#)
- ❖ [Kafka Training](#)
- ❖ [Kafka Consulting](#)