```
public class Account extends Identity {
    @NotNull
    @SamplePath(field = SampleFieldId.COMPANY, readable =
"account.company")
    private Company company;
}

public enum SampleFieldId implements SampleField {
    TIMEZONE(ACCOUNT),
    COUNTRY(ACCOUNT),
    COMPANY(ACCOUNT),
}

public class RulesOld {

    public static boolean validateAccount(User user, …) {
        if (config == null) {
            return false;
        }
        if (user == null || user.getBirthDate() == null) {
            return false;
        }
        if (account == null || account.getCountry() == null ||
            account.getPhoneNumber() == null) {
            return false;
        }
        if (YEARS.between(user.getBirthDate(), LocalDate.now()) >= 18
                && account.getEmail().length() <= config.getMaxEmailSize()
                && account.getCompany() == Company.LES_FURETS
                && account.getPhoneNumber().startsWith("+33")) {
            return true;
        }
        return false;
    }

}
```

```java
public class RulesOpenRndayTest {
    SampleModel sample = SampleModels.sample();
    /**
     * @see RulesOld#validateAccount()
     */
    SampleModelRule demoRule = DslSampleModel.when(DOOV.matchAll(
            userBirthdate.ageAt(today()).greaterOrEquals(18),
            accountEmail.length().lesserOrEquals(configurationMaxEmailSize),
            accountCompany.eq(Company.LES_FURETS),
            accountPhoneNumber.startsWith("+33")))
      .validate();

    @Test
    public void test_account() {
        Result result = demoRule.executeOn(sample);
        System.out.println(demoRule.readable());
        System.out.println(demoRule.markdown(Locale.FRANCE));
        Assertions.assertThat(result).isTrue();
}

    @Test
    public void test_account_failure_cause() {
        sample.getAccount().setPhoneNumber("+1 12 34 56 78");
        Result result = demoRule.executeOn(sample);
        System.out.println(result.getFailureCause());
        Assertions.assertThat(result)
                .isFalse()
                .hasFailureCause("account phone number starts with '+33'");
    }

    @Test
    public void test_account_failure_cause_2() {
        sample.getAccount().setPhoneNumber("+1 12 34 56 78");
        sample.getAccount().setCompany(Company.BLABLACAR);
        Result result = demoRule.withShortCircuit(false)
                                .executeOn(sample);
        System.out.println(result.getFailureCause());
        Assertions.assertThat(result)
                .isFalse()
                .hasFailureCause("match all [" +
                        "account company = LES_FURETS, " +
                        "account phone number starts with '+33']");
    }
}
```

1. show SampleModel
2. show Account
3. show annotation + readable
4. add company field
5. add annotation
6. add field id
7. show RessourceBundle/property file
8. run gradle -p sample build -x test

9. show DslSampleModel
10. review legacy RuleOld
11. add predicate about account.company == LES_FURETS
12. rewrite RuleOld with DSL

13. open RulesOpenRndayTest
14. review SampleModels.sample()
15. add a field value for account.company

16. add asserThat(result).isTrue()
17. add sysout(rule.readable())
18. add sysout(rule.markdown(Local.FR))
19. run

20. delete previous sysout
21. duplicate the test
22. add wrong phone number
23. run

24. review the test failure trace
25. add sysout(result.getFailureCause())
26. change assert and assertFailureCause

27. delete previous sysout
28. duplicate the test
29. add wrong company
30. add sysout(result.getFailureCause())
31. run

32. add executeOn().withShortCircuit(false)
33. run

34. show the lesfurets/maintenance/exclusion

35. run BenchmarkUI