

LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

**DOMĒNA SPECIFISKĀ VALODA MOODLE TESTU
IZVEIDOŠANAI**

BAKALaura DARBS

Autore: **Jevgeņija Babčenoka**

Studenta apliecības Nr.:jb19045

Darba vadītāja: Dr.dat. Elīna Kalniņa

RĪGA 2023

ANOTĀCIJA

Mūsdienās Moodle ir populārākā mācību platforma. Arvien vairāk pasniedzēju izmanto Moodle testus studentu zināšanu pārbaudei. Bakalaura darba ietvaros tika izstrādāta domēna specifiskā valoda Moodle testu izveidošanai. Testa jautājumi tiek atlasīti no kursa jautājuma bankas. Lai izveidotu domēna specifisko valodu, tika izmantots JetBrains MPS rīks. Tāpat tika izstrādāts jauns Moodle spraudnis domēna specifiskās valodas interpretatoram.

Atslēgvārdi: domēna specifiskā valoda, JetBrains MPS, Moodle, Moodle spraudnis.

ABSTRACT

DOMAIN SPECIFIC LANGUAGE FOR CREATING MOODLE TESTS

Moodle is the most popular learning platform today. Instructors are increasingly using Moodle quizzes to test student`s knowledge. As part of the bachelor's thesis, a Domain-Specific language was developed for creating Moodle quizzes. Questions are selected from the course question bank. The JetBrains MPS tool was used to create the Domain-Specific language. Additionally, the new Moodle plugin was developed for the Domain-Specific language interpreter.

Keywords: Domain-Specific language, JetBrains MPS, Moodle, Moodle plugin.

SATURA RĀDĪTĀJS

APZĪMĒJUMU SARAKSTS	5
IEVADS	6
1. MOODLE.....	8
1.1. Moodle sastāvdaļas	8
1.2. Datu bāze	10
1.2.1. Datu bāzes struktūra	10
1.2.1. Piekļuve datu bāzei	11
2. DOMĒNA SPECIFISKĀS VALODAS	13
2.2. DSL veidi	14
2.3. DSL izstrādes rīki	14
2.2.1. Eclipse Foundation	15
2.2.2. MetaEdit+	16
2.2.3. JetBrains MPS	16
2.4. DSL mācību pārvaldības sistēmās	16
3. JETBRAINS MPS	17
3.2. DSL definēšanas modulis	18
3.1.1. Struktūra (Structure)	18
3.1.2. Redaktors (Editor)	18
3.1.3. Ierobežojumi (Constraints)	19
3.1.4. Uzvedība (Behavior)	19
3.1.5. Ģenerators (Generator)	19
3.1.6. TextGen	19
4. DSL MOODLE TESTU IZVEIDOŠANAI	20
4.1. DSL domēns	20
4.2. DSL sintakse	22
4.3. DSL definēšanas modulis	23
4.3.1. DSL struktūras aspekts	24

4.3.2. DSL redaktora aspekts	27
4.3.3. DSL ģeneratora aspekts.....	29
4.4. Moodle spraudnis.....	33
4.4.1. Spraudņa izveidošanas process	33
4.4.2. DSL interpretatora algoritms.....	33
4.5. DSL izmantošanas piemērs.....	36
REZULTĀTI UN DISKUSIJA.....	41
SECĪNĀJUMI	42
PATEICĪBAS.....	44
IZMANTOTĀ LITERATŪRA UN AVOTI.....	45
PIELIKUMI	48
1. pielikums. DSL koncepti.....	48
2. pielikums. DSL konceptu redaktori	52
3. pielikums. DSL pārveidošanas veidnes.....	54
4. pielikums. Kursa jautājumu bankas piemērs.....	57
5. pielikums. Uzģenerētā XML faila saturs	59
6. pielikums. Testa 2. varianta uzdevumi.....	61
7. pielikums. Testa 3. varianta uzdevumi.....	62
8. pielikums. Testa 4. varianta uzdevumi.....	63
9. pielikums. Izstrādātā XML faila augšupielādes forma	64
10. pielikums. Jautājumu atlasīšana, izmantojot nosaukuma apakšvirkni.....	65
11. pielikums. Jautājumu atlasīšana, izmantojot tagu	66

APZĪMĒJUMU SARAKSTS

AST (*Abstract Syntax Tree*) – abstraktais sintakses koks.

BNF (*Backus-Naur Form*) – formālā gramatika formālās valodas sintakses definēšanai.

DSL (*Domain Specific Language*) – domēna specifiskā valoda.

EBNF (*Extended Backus-Naur Form*) – paplašināta BNF versija.

GPL (*General-purpose programming language*) – vispārējā programmēšanas valoda.

IDE – integrētā izstrādes vide.

JavaScript – interpretētā programmēšanas valoda tīmekļa lietojumprogrammu izstrādei.

JetBrains MPS – integrētā valodu izstrādes vide.

LMS (*Learning management system*) – mācību pārvaldības sistēma.

Moodle – mācību pārvaldības sistēma.

PHP – servera puses programmēšanas valoda tīmekļa lietojumprogrammu izstrādei.

SQL (*Structured Query Language*) – valoda, kas tiek izmantota datu bāzu pārvaldībai.

XML (*Extensible Markup Language*) – paplašināmā iezīmēšanas valoda.

IEVADS

Mūsdienās izglītības jomā arvien vairāk izmanto mācību pārvaldības sistēmas (*Learning Management System – LMS*), kas paredzētas mācību procesa atbalstīšanai digitālajā vidē. LMS tiek plaši izmantotas skolās, universitātēs un uzņēmumos cilvēku izglītībai un apmācībai. Īpaši mācību pārvaldības sistēmas kļuva populāras COVID-19 pandēmijas dēļ. Viens no galvenajiem LMS elementiem ir kursu pārvaldība, kas ļauj pārvaldīt to saturu – publicēt kursu materiālus, interaktīvus uzdevumus, sniegt vērtējumus, atgriezeniskās saites utt. Šobrīd eksistē daudz dažādu mācību pārvaldības sistēmu. Piemēram, Moodle [1], Blackboard Learn [2], Canvas [3] mācību pārvaldības sistēma. Vispopulārākā mācību pārvaldības sistēma Latvijā ir Moodle [4].

Viena no galvenajām Moodle funkcionalitātēm ir interaktīvo testu mehānisms. Pārsvārā Moodle interaktīvie testi tiek izmantoti, lai pārbaudītu lietotāju zināšanas un prasmes. Īpaši tie tiek izmantoti pārbaudījumiem un eksāmeniem. Pirms testu izveidošanas pasniedzējs izveido jautājumus, kuri saglabājas kursa jautājumu bankā. Lai izveidotu testu, pasniedzējs izvēlas eksistējošos jautājumus no jautājumu bankas un pievieno tos testā. Katram testam pasniedzējs pievieno dažādus iestatījumus. Piemēram, mēģinājumu skaitu un piekļuves ierobežojumus. Šis process var aizņemt ilgu laiku. Līdz ar to testu izveidošanas un jautājumu atlases procesu var paātrināt un automatizēt, izmantojot domēna specifisko valodu. Bakalaura darba ietvaros tika izstrādāta:

- domēna specifiskā valoda Moodle testu izveidošanai;
- jauns Moodle spraudnis testu izveidošanai.

Izstrādātā domēna specifiskā valoda ļauj:

- izveidot vairākus testus un to variantus;
- atlasīt jautājumus no Moodle kursa jautājumu bankas, izmantojot nosaukumu apakšvirknes vai tagus;
- sadalīt atlasītos jautājumus variantos tā, lai jautājumi variantos atšķirtos;
- pievienot iestatījumus katram testa variantam.

Domēna specifiskā valoda tika realizēta, izmantojot JetBrains MPS rīku. JetBrains MPS rīkā tika izveidots valodas definēšanas modulis, kas ietver valodas struktūras, redaktora un ģeneratora aspektu. Ģeneratora aspektā tika definēti domēna specifiskās valodas modeļu transformēšanas likumi XML formāta modeļos. Līdz ar to ir iespējams ģenerēt XML dokumentu, balstoties uz ievadīto domēna specifiskās valodas kodu.

XML turpmākai apstrādei tika izstrādāts Moodle spraudnis. Jaunais spraudnis satur XML augšupielādes formu un domēna specifiskās valodas interpretatoru. Domēna specifiskās

valodas interpretators izveido vairākus testus un to variantus kursā, balstoties uz XML dokumenta saturu. Tā kā bakalaura darba ietvaros tika izstrādāts domēna specifiskās valodas un Moodle spraudņa prototips, tas var tikt paplašināts un uzlabots.

Bakalaura darba pirmajā nodaļā ir aprakstīta Moodle mācību pārvaldības sistēmas nozīme, arhitektūra un datu bāzes struktūra. Otrajā nodaļā ir aprakstīta domēna specifiskās valodas definīcija, domēna specifisko valodu veidi un izstrādes rīki. Trešā nodaļa satur JetBrains MPS rīka aprakstu un tās galvenos aspektus. Ceturtajā nodaļā ir aprakstīta izstrādātā domēna specifiskā valoda, jaunais Moodle spraudnis un viens domēna specifiskās valodas izmantošanas piemērs.

1. MOODLE

Mūsdienās arvien vairāk izmanto attālinātās mācības, kas ietekmē dažādu mācību digitālo rīku attīstīšanu. Viens no digitāliem rīkiem, kas palīdz kombinēt klātienes mācības ar attālinātām ir mācību pārvaldes sistēma. Eksistē daudz dažādu mācību pārvaldes sistēmu un viena no tām ir Moodle [1]. Moodle platforma tiek plaši izmantota dažādās skolās, universitātēs un uzņēmumos. Tostarp Moodle ir populārākā mācību pārvaldības sistēma Latvijā. Piemēram, Latvijas Universitātes (LU), Rīgas Tehniskās Universitātes (RTU) ORTUS un Rīgas Stradiņņas universitātes (RSU) e-studiju vide tiek bāzēta uz Moodle mācību pārvaldības sistēmas [4]. Tādēļ Moodle platforma tika izvēlēta kā platforma domēna specifiskās valodas izveidei bakalaura darba ietvaros.

Moodle platformas izstrādātāji arvien vairāk uzlabo un papildina Moodle ar jaunu funkcionalitāti. Pašlaik pasniedzēji var pārvaldīt mācību procesu Moodle platformā. Piemēram, pievienot mācību materiālus, sazināties ar kursu dalībniekiem, veidot interaktīvos testus, sniegt vērtējumus un atsauksmes. Savukārt studenti var izmantot šos materiālus zināšanu un prasmju apguvei. Moodle sistēma satur daudzas funkcijas, kas noderīgas veiksmīgam mācību procesam [1].

Moodle mācību pārvaldības sistēma ir atvērta pirmkoda programmatūra, kas nozīmē, ka sistēmas kods ir publiski pieejams. Moodle platformas pirmkods ir pieejams publiskajā repozitorijā [5]. Tāpēc izstrādātāji var brīvi izmantot un modificēt Moodle sistēmas kodu, kā arī pielāgot sistēmu saviem izglītības mērķiem. Ir iespējams izstrādāt savus sastāvdaļas spraudņu veidā, kas paplašina platformas funkcionāli.

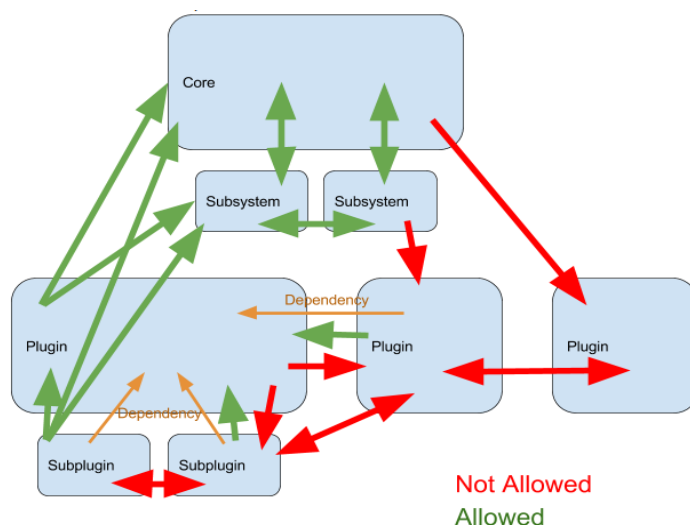
Moodle pirmkods pārsvarā ir rakstīts PHP programmēšanas valodā [6] [7]. Tikai daļēji tika izmantota JavaScript programmēšanas valoda [8], Gherkin, CSS un citi rīki [6]. Līdz ar to jaunās Moodle spraudņa izveidei (sk. 4.4. apakšnodaļu) tika izmantota PHP programmēšanas valoda.

Sakarā ar to ka bakalaura darba ietvaros tiek izstrādāta domēna specifiskā valoda un jauns Moodle spraudnis, ir jāsaprot tās arhitektūra. Līdz ar to tika aprakstītas Moodle sastāvdaļas (sk. 1.1. apakšnodaļu) un Moodle datu bāzes struktūra (sk. 1.2. apakšnodaļā).

1.1. Moodle sastāvdaļas

Moodle pamatā ir modulārā arhitektūra, kura ļauj pievienot un pielāgot dažādus platformas moduļus. Moodle dokumentācijā tika minēts, ka Moodle arhitektūra tiek veidota no vairākām “sastāvdaļām” (*Components*). Sastāvdaļas un to atkarības var redzēt 1.1. attēlā [6].

Tāpat 1.1. attēlā parādīti Moodle sastāvdaļu komunikācijas ierobežojumi, kas atzīmēti ar sarkanām bultām.



1.1. att. – Moodle sastāvdaļu atkarības un komunikācijas ierobežojumi [9]

Moodle platformas pamata sastāvdaļa ir lietojumprogrammas kodols (*Moodle Core*) (sk. 1.1. attēlu) [9]. Lietojumprogrammas kodols satur galvenās funkcijas, kas ir nepieciešamas platformas veiksmīgai darbībai. Piemēram, datu apstrāde datu bāzē, kešatmiņas nodrošināšana, datubāzes lietotāju, lietotāju piekļuves, kursu un to saturu pārvaldība, pieteikšanās kursiem utt. [6] Moodle lietojumprogrammas kodols atrodas “/lib/” pirmkoda mapē. Šīs funkcijas ir vienmēr pieejamas jebkurā Moodle platformas vietā un nevar tikt atslēgtas vai dzēstas.

Kā arī Moodle lietojumprogrammas kodols satur apakšsistēmas (*Subsystems*) (sk. 1.1. attēlu). Lietojumprogrammas apakšsistēmas satur ne tikai galvenās funkcijas, bet arī nodrošina konkrētu funkcionalitāti. Piemēram, komunikācijas apakšsistēma. Katra lietojumprogrammas kodola apakšsistēma nodrošina papildus funkcionalitāti. Atšķirībā no lietojumprogrammas kodola pamata funkcijām, platformas administrators tīmekļa lietojumprogrammas administrēšanas sadaļā var atspējot vai iespējot jebkuru lietojumprogrammas kodola apakšsistēmu [9].

Moodle funkcionalitāti var paplašināt, izmantojot papildus spraudņus (*Plugin*) un apakšspraudņus (*Sub-plugins*) (sk. 1.1. attēlu). Moodle platformas izstrādātāji var veidot savus spraudņus saviem mērķiem vai pievienot eksistējošus. Piemēram, Moodle vietnē (Moodle spraudņu katalogā [10]) tiek piedāvāti publiskie spraudņi, kas var tikt pieslēgti mācību vidē. Būtībā jebkurš Moodle spraudnis ir PHP [7], JavaScript [8], CSS un citu rīku skriptu kopa. Moodle spraudnis var būt atkarīgs no cita spraudņa (sk. 1.1. attēlu) un šo atkarību definē spraudņa “version.php” failā [9]. Visus spraudņus ir jāizstrādā Moodle pirmkoda mapē “local”.

1.2. Datu bāze

Moodle datu bāze ir svarīgs Moodle arhitektūras elements. Izmantojot datu bāzi, tiek nodrošināta datu pārvaldība – visas *CRUD* (*create, read, update and delete - CRUD*) operācijas (datu saglabāšana, atjaunošana, iegūšana un dzēšana). Moodle izmanto relāciju datu bāzi, kas nozīmē, ka dati tiek glabāti tabulās kolonnu un rindu veidā.

Pirms Moodle instalēšanas ir jāizvēlas datu bāze, kur Moodle saglabās savus datus. Moodle datu bāze var būt MySQL, PostgreSQL, Microsoft SQL Server, MariaDB vai Oracle [6]. Tīmekļa lietojumprogrammu datu bāzes migrācijas gadījumā (pārnest datus no vienas datu bāzes uz otru) administrators var izmantot datu bāzes pārsūtīšanas rīku (*database transfer tool*) [11].

1.2.1. Datu bāzes struktūra

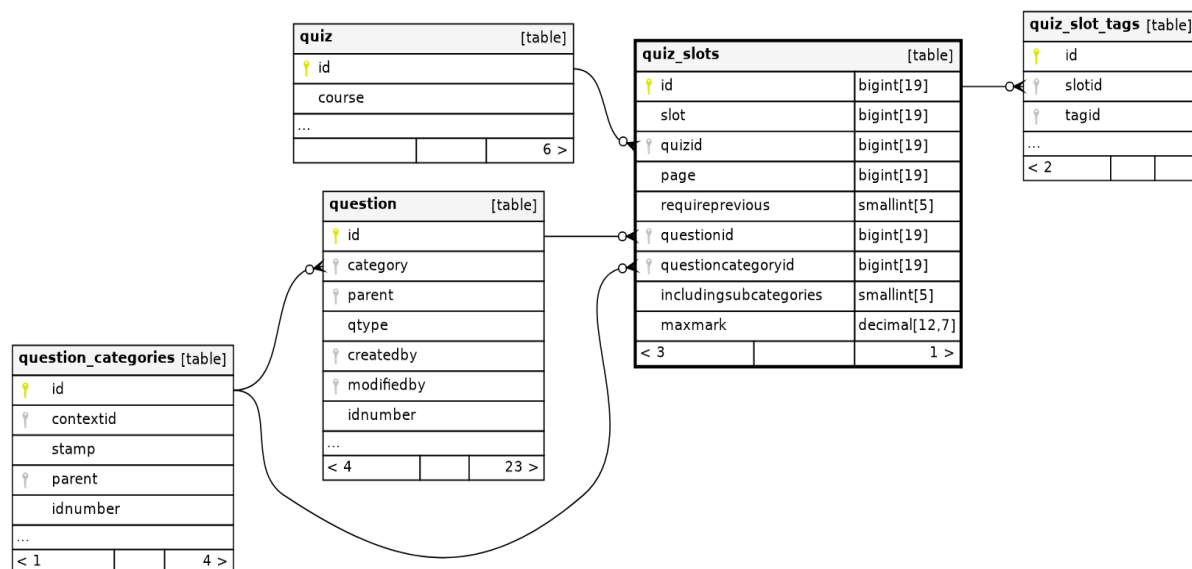
Tā kā viens no galvenajiem bakalaura darba mērķiem ir izstrādāt domēna specifisko valodu Moodle testu izveidošanai, šajā apakšnodaļā parādīta datu bāzes daļa, kas ir saistīta ar interaktīviem testiem un testu jautājumiem. Datu bāzes struktūras saprašana palīdz izveidot spraudni, kas aprakstīts 3. nodaļā. Piemēram, ir nepieciešams saprast kā ir iespējams iegūt no datu bāzes kursu, testa jautājumus un iestatījumus. Pilnu datu bāzes struktūru var redzēt datu bāzes shēmā. Datu bāzes shēma apraksta datu struktūru un organizāciju datu bāzē (parāda visas tabulas, laukus un attiecības starp tabulām). Tomēr Moodle datu bāzes struktūra ir atkarīga no Moodle versijas. Tāpēc katrai Moodle versijai ir pieejama atsevišķa datu bāzes shēma. Katras Moodle versijas datu bāzes shēmu var atrast Moodle dokumentācijā [6]. Bakalaura darba ietvaros tiek izmantota Moodle 4.0. versijas datu bāzes shēma, kurā ir vairāk nekā 450 entītijas [12].

Ir nepieciešams apskatīt 4 (četras) entītijas, kas noderēs, izstrādājot jauno Moodle spraudni:

- “Quiz” (sk. 1.3. attēlu);
- “Quiz_slots” (sk. 1.2. attēlu);
- “Question” (sk. 1.2. attēlu);
- “Question_category (sk.1.2. attēlu)”.

“Quiz” (sk. 1.3. attēlu) entītija satur interaktīvo testu datus: nosaukumu, saistīto kursu, pieejamības laiku, paroli, mēģinājumu skaitu, vērtējumu iestatījumus utt. [13] Šo entītiju var redzēt 1.3. attēlā. Tāpat var redzēt, ka “Quiz” entītija satur atsauces uz citām entītijām. Viena no tām ir “Quiz_slots”. “Quiz_slots” glabā datus par katru jautājumu testā – jautājuma identifikatoru un to pozīciju testā. “Quiz_slots” entītiju var redzēt 1.2. attēlā [14]. Katrs testa

jautājums tiek glabāts “Question” entītijā (sk. 1.2. attēlu). Tāpat arī ir svarīgi zināt, ka katrs jautājums satur atsauci uz jautājuma kategoriju, kas glabājas “Question_category” entītijā. Visus kursa jautājumus un to kategorijas lietotājs var apskatīt kursa jautājumu bankā. Tāpat arī jautājumu bankā lietotājs var pārvaldīt jautājumus un to kategorijas.



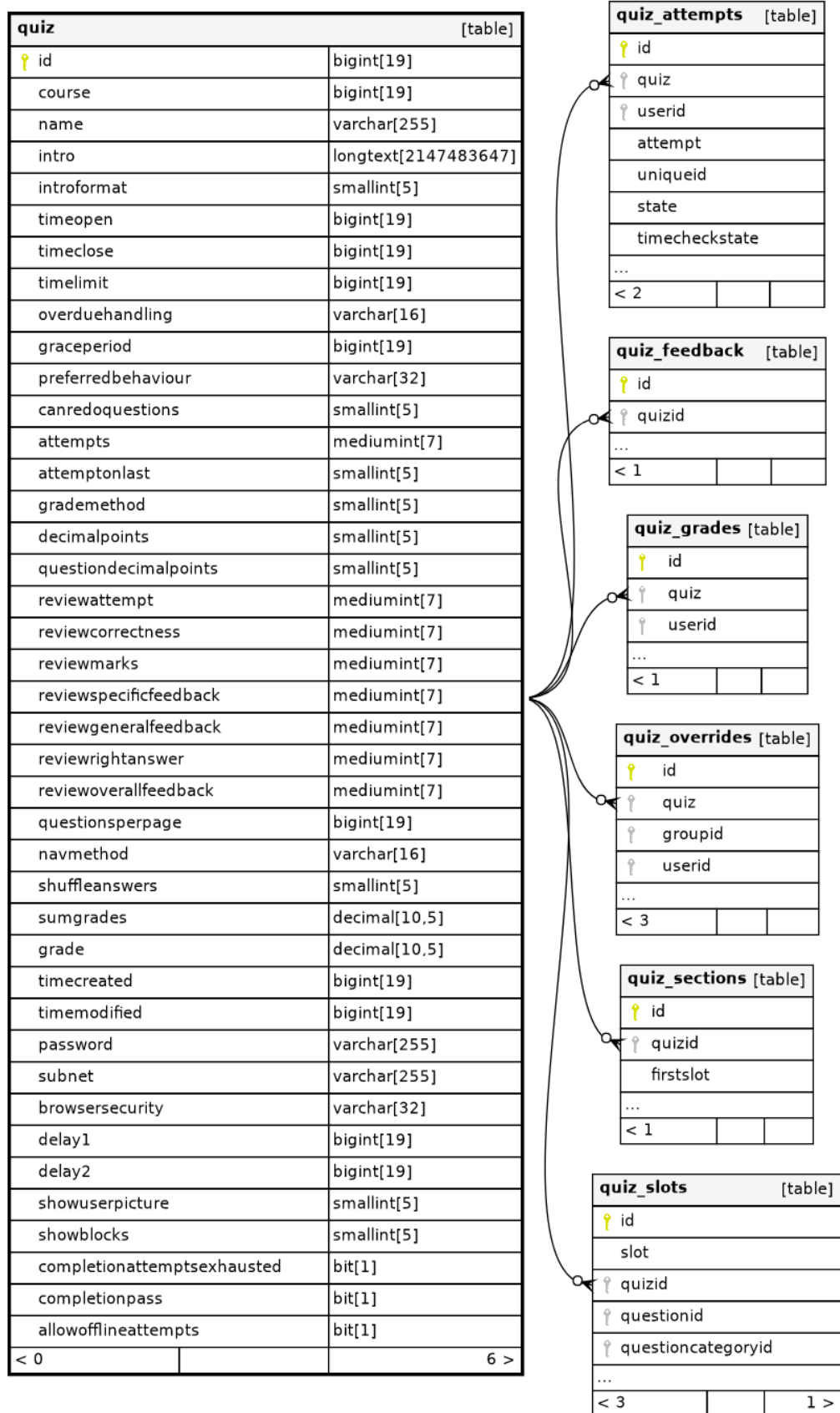
1.2. att. – Moodle “quiz_slots” entīcija [14]

1.2.2. Piekļuve datu bāzei

Eksistē funkcijas, kuras izstrādātāji var izmantot, lai piekļūtu datu bāzei un manipulēt ar tajā saglabātajiem datiem. Šādas funkcijas ir aprakstītas Moodle dokumentācijā, datu manipulācijas API (*Data Manipulation API*) sadaļā.

Viens no galvenajiem Moodle datu bāzes elementiem ir globāls \$DB objekts. Globāls \$DB objekts (moodle_database klases instance) tiek izmantots Moodle datubāzes piekļūšanai un datu manipulēšanai. Tas nodrošina ērtu vaicājumu izpildi objekt-orientētā veidā. \$DB objekts tiek automātiski inicializēts sistēmas palaišanas laikā un var būt pieejams jebkurā Moodle platformas vietā – gan visos moduļos, gan visos spraudņos [15].

Lai izmantotu globālo \$DB objektu, ir jānodrošina savienojums ar datu bāzi. Failos, kuros ir jāizmanto datu bāze, ir jānodrošina “config.php” faila iekļaušana. “config.php” failā atrodas Moodle datu bāzes pieslēgšanas konfigurācija.



1.3. att. – Moodle “Quiz” entităte [13]

2. DOMĒNA SPECIFISKĀS VALODAS

Viena no programmatūras izstrādes pieejām ir domēna specifiskās valodas (*Domain-Specific language - DSL*) izmantošana. Domēna specifiskā valoda ir valoda, kas paredzēta specifiskām domēnam. Izvēloties konkrēto domēnu un sašaurinot problēmas apgabalu, var ievērojami vienkāršot problēmas risināšanu. Domēna specifisko valodu ir iespējams izstrādāt, izceļot galvenos domēna jēdzienus.

Musdienās eksistē daudz vispārējo programmēšanas valodu (*general-purpose programming language - GPL*), kuras izmanto daudzveidīgiem uzdevumiem. Salīdzinot ar GPL, DSL nodrošina augstāku abstrākcijas līmeni, kā arī palīdz izteikt problēmu īsāk un lakoniskāk. DSL nav universāla valoda, bet precīzi apraksta un risina problēmu. Konkrētajos domēnos DSL nodrošina labāku efektivitāti.

Daži nevar saprast, kāpēc ir nepieciešams izstrādāt jaunu DSL, ja domēna problēmu var izteikt, piemēram, izmantojot XML vai JSON formātā. Protams, ka šajā veidā var izteikt un risināt daudzveidīgas problēmas. Tomēr parastājiem lietotājiem XML un JSON formātu grūti ne tikai izmantot, bet arī lasīt, jo tie ir ļoti “mašīnlasāmi” formāti [16]. Informāciju, kuru var pierakstīt XML un JSON formātos, var arī pierakstīt lietotājiem “draudzīgākā” formātā. Līdz ar to DSL var atvieglot problēmas risināšanu.

DSL tiek izstrādāts, lai to izmantotu domēna speciālisti, kuri pārzina specifisko domēnu. Parasti domēna speciālistiem trūkst programmēšanas prasmju, lai izmantotu vispārējo programmēšanas valodu. Lai domēna speciālistiem atvieglotu DSL izmantošanu, tas tiek izstrādāts, ņemot vērā domēna specialistu ieteikumus. Lai izstrādātu DSL, ir nepieciešams definēt risināmo problēmu un izcelt galvenus konceptus. Pēc tām DSL izveidošanas procesu var sadalīt trīs daļās:

- Abstraktās sintakses definēšana;
- Konkrētās sintakses definēšana;
- Transformāciju likumu definēšana.

Abstraktā sintakse apraksta valodas jēdzienus, to īpašības un savstarpējo mijiedarbību. Abstrakto sintaksi ir iespējams definēt, izmantojot meta-modeli vai gramatiku (tas atkarīgs no domēnas specifiskās valodas veida). Konkrētā sintakse definē abstraktās sintakses attēlojumu, kas var būt tekstuālā un/vai grafiskā veidā. Savukārt transformāciju likumi satur likumus, pēc kuriem abstraktā sintakse tiek pārveidotā izpildāmā kodā.

2.2. DSL veidi

Eksistē dažādas domēna specifiskās valodas. Domēna specifiskās valodas veids tieši ietekmē gan tās izstrādi, gan tās lietošanu. Piemēram, sākot izstrādāt DSL, ir jāsaprot, kurš DSL izstrādes rīks (sk. 2.3. apakšnodaļu) būtu vispiemērotākais tā izstrādei.

Pirmkārt, ir tekstuālās un grafiskās domēna specifiskās valodas. Atšķirībā no tekstuālā DSL, grafiskā DSL konkrētā sintakse satur grafiskos elementus. Otrkārt, domēna specifiskās valodas var sadalīt divās citās grupās: iekšējās un ārējās domēna specifiskās valodas [4].

Iekšējo DSL (*Internal DSL*) daži sauc par iegulto DSL (*Embedded DSL*). Iekšējā domēna specifiskā valoda tiek izstrādāta tā, lai tās izpildītu instrukciju virknes GPL valodā. Tās visbiežāk tiek izstrādātās kā GPL valodas ietvars vai bibliotēka [17]. Tomēr iekšējam DSL ir trūkums – tas ir atkarīgs no GPL valodas sintakses. Sintakses neatbilstības gadījumā kompilators nevarēs veikt sintaktisko analīzi, parsēt un ģenerēt kodu, balstoties uz izstrādāto domēna specifisko valodu [18].

Ārējais (*External*) DSL ir savrups DSL, kas ir atdalīts no GPL sintakses. Ārējam DSL ir nepieciešams būvēt savu interpretatoru vai kompilatoru, kas balstās uz citām GPL valodām [17]. Viena no populārākajām ārējām domēna specifiskām valodām ir SQL (*Structured Query Language*) valoda, kas paredzēta datu bāzes pārvaldībai. Kā jau var saprast, SQL domēns – darbības relāciju datubāzēs. Viens no ārējā DSL trūkumiem ir tāds, ka tas ir izolēts. Otrs trūkums ir tāds, ka ārējam DSL nav IDE vides. Ir nepieciešams izstrādāt speciālo redaktoru, kur būs iespējams izmantot izstrādāto valodu. Kā arī, lai izmantotu DSL, ir nepieciešams izstrādāt DSL importa un/vai eksporta funkcionalitāti [18].

2.3. DSL izstrādes rīki

Eksistē daudz dažādu rīku, kuros ir iespējams izstrādāt domēna specifiskās valodas. Eksistējošie rīki ļauj definēt DSL abstrakto un/vai konkrēto sintaksi, ģeneratoru, redaktoru utt. Sākot domēna specifisko valodas izstrādi, ir jāņem vērā tās veids (sk. 2.2. apakšnodaļu), mērķis un paredzēto izmantošanas scenāriju. Tas tieši ietekmē DSL izveidošanas procesu. Šajā apakšnodaļā ir aprakstīti vairāki populārākie DSL izstrādes rīki (sk. 2.2.1. – 2.2.3. apakšnodaļu).

2.2.1. Eclipse Foundation

Eclipse Foundation ir bezpeļņas organizācija, kas izstrādā un uztur atvērtā koda programmatūru. Viens no galvenajiem Eclipse Foundation projektiem ir Eclipse IDE. Tomēr Eclipse Foundation pārvalda dažādus projektus, tostarp izstrādāja vairākus rīkus DSL izstrādei [19]. Piemēram:

- Eclipse Modeling Framework (EMF);
- Eclipse Sirius;
- Graphical Modeling Framework (GMF);
- Xtext Eclipse Foundation.

Eclipse Modeling Framework (EMF) ir ietvars modeļu izstrādei. EMF rīks ļauj izveidot strukturētus datu modeļus – metamodeļus, kurus EMF sauc par Ecore. Tāpat EMF nodrošina daudzus rīkus darbam ar Ecore modeļiem – Java klases ģenerēšanu, redaktorus un citus rīkus [20]. EMF var integrēt arī ar citiem Eclipse rīkiem – Eclipse IDE, GMF utt. Līdz ar to, izstrādātāji var izmantot EMF kopā ar citiem rīkiem un ietvariem, lai izveidotu jaudīgas lietojumprogrammas un modelēšanas rīkus.

Eclipse Sirius ir rīks, kurā ir iespējams izveidot grafiskās modelēšanas rīkus balstoties uz iepriekš definētiem metamodeļiem. Eclipse Sirius rīks izmanto EMF definētus modeļus kā pamatu vizuālo redaktoru un modelēšanas rīku izveidei. Šo modelēšanas rīku plaši izmanto sistēmu un programmatūras inženierijā. Sirius vietnē tika publicētas populārākās sistēmas, kas ir izstrādātas, izmantojot Sirius platformu [21]. Piemēram, tika izstrādāts Midstorms Designer rīks, kas ļauj aprakstīt Lego Midstorms robota darbības un ģenerēt Java kodu, kuru vēlāk var izpildīt Lego Midstorms robotā [22].

Graphical Modeling Framework (GMF) ir ietvars, kas ļauj izveidot grafiskos redaktorus, pamatojoties uz EMF ietvara metamodeļiem. Tomēr GMF pašlaik ir novecojis un tā vietā tiek izmantots Eclipse Sirius rīks.

Xtext ir populārs Eclipse ietvars, kas ļauj izstrādāt tekstuālo DSL [23]. Xtext piedāvā daudz vairāk funkciju nekā parastais parseris. Izstrādātājs var definēt ne tikai parseri, bet arī kodu ģenerēšanu un modeļu transformāciju. Xtext rīkā DSL tiek veidots, sākmā definējot tekstuālo sintaksi. Tekstuālā sintakse tiek definēta, izmantojot EBNF (*Extended Backus-Naur Form*) formālo apzīmējumu, kas ir BNF (*Backus-Naur Form*) paplašinātā versija. EBNF ir veids, kā var definēt formālās valodas gramatiku un aprakstīt valodas sintaksi. Pēc tekstuālās sintakses definēšanas Xtext ietvars pārveido definēto sintaksi abstraktās sintakses koka (*Abstract Syntax Tree - AST*) klašu modelī [24].

2.2.2. MetaEdit+

MetaEdit+ ir domēna specifiskā modelēšanas vide, ko izstrādāja MetaCase uzņēmums. Šis rīks ļauj izveidot un izmantot domēna specifiskās modelēšanas valodas (*Domain-Specific modeling languages - DSMLs*). Lai definētu DSML, izstrādātājs definē metamodeli, izmantojot MetaEdit+ GOPRR metamodelēšanas valodu. MetaEdit+ ļauj noteikt DSML konceptus, likumus un notācības. Kā arī MetaEdit+ rīkā var definēt ģeneratoru – kodu, kas tiks uzģenerēts no modeļiem. Eksistē divas MetaEdit+ versijas: MetaEdit+ Workbench un MetaEdit+ Modeler. Valodas definīcija saglabājas MetaEdit+ repositoriņā, kuru turpmāk var koplietot [25]. MetaCase ir oficiāla vietne, kurā ir aprakstīti MetaCase rīki un MetaCase+ izstrādātie DSM [26].

2.2.3. JetBrains MPS

JetBrains MPS (*Meta Programming System*) ir integrētā izstrādes vide domēna specifisko valodu izstrādei (*Language Workbench*) [27]. JetBrains MPS ir viens no jaudīgiem un plašāk izmantotiem DSL rīkiem pasaulē. Šis rīks atšķiras no citiem rīkiem ar to, ka kods šajā valodas integrētā izstrādes vidē glabājas abstraktā sintakses kokā [28]. Bakalaura darba ietvaros tika izmantots tieši šis rīks DSL Moodle testu izveidošanai. Sakarā ar to JetBrains MPS rīks tiks aprakstīts detalizētāk 3. nodaļā.

2.4. DSL mācību pārvaldības sistēmās

Eksistē vairākas tekstuālās un grafiskās domēna specifiskās valodas, kas paredzētas mācību pārvaldības sistēmām. Piemēram, tika izstrādāta SASQL domēna specifiskā valoda kompetenču novērtējumu pielāgošanai [29]. SASQL izmantošanai tika izstrādāta EvalCourse darbvirsmas lietojumprogramma, kura komunicē ar Moodle mācību pārvaldības sistēmu datu iegūšanai no darbību žurnāliem. Otrkārt, tika izstrādāts KiwiDSM rīks, kurš satur grafisko domēna specifisko valodu mācību pārvaldības sistēmu moduļu ģenerēšanai [30]. KiwiDSM rīks ļauj veidot mācību pārvaldības sistēmu moduļus grafiskā veidā vairākām mācību pārvaldības sistēmām. Tāpat tika izstrādāts AdaptiveVLE mācību personalizēšanas ietvars [31]. AdaptiveVLE satur domēna specifisko valodu ietvara konfigurēšanai un CAF (*Classification Algorithms Framework*) domēna specifisko valodu turpmākai mācību personalizēšanai. Aprakstītas domēna specifiskās valodas tika aprakstītas detalizētāk kursa darbā [4].

3. JETBRAINS MPS

JetBrains MPS (Meta Programming System) – ir atvērta pirmkoda integrētā valodu izstrādes vide (*Language Workbench*) [27]. JetBrains MPS rīkā ir iespējams projektēt, izveidot domēna specifiskās valodas (*domain specific languages – DSLs*). JetBrains rīks atbalsta dažādas programmēšanas valodas, piemēram: Java, JavaScript [8], C# un XML (*Extensible Markup Language*) iezīmēšanas valodu. JetBrains MPS ir viens no jaudīgākiem un plašāk izmantotiem DSL rīkiem pasaulē un to izmanto dažādos domēnos. Piemēram, JetBrains MPS rīks tiek pielietots medicīnā, datu zinātnē, robotikā, telekomunikācijas un finanšu jomā [32].

JetBrains MPS atšķirās no citiem DSL rīkiem. Tradicionālos DSL rīkos tiek izmantots kompilators, kas izmanto lekseri un parseri, lai pārveidotu programmas tekstu abstraktā sintakses kokā (*Abstract Syntax Tree – AST*). Kompilators ģenerē izpildāmo kodu, balstoties uz AST. Savukārt JetBrains MPS izvairās no tekstuālās formas – valodas darbvirsma programmas kods tiek automātiski ģenerēts, saglabāts, atjaunināts un analizēts abstraktā sintakses kokā. Lietotājs strādā tieši ar AST. Šo pieeju sauc par projekcijas rediģēšanu (*projectional editing*) [32] [28].

Projekcijas rediģēšanai ir daudz priekšrocību. Piemēram, projekcijas rediģēšana atvieglo DSL izveidošanas procesu, jo nav nepieciešams būt gramatikai un parsētājam. Otra priekšrocība ir tāda, ka kodā ir iespējams izmantot ne tikai tekstu, bet arī diagrammas, tabulas, formas un citas līdzīgas lietas. Trešā priekšrocība ir iespēja izmantot vairākas DSL vienā programmā [32] [33].

AST JetBrains MPS vidē nodrošina augsta līmeņa skatu uz koda struktūru, tādējādi atvieglo DSL veidošanu. AST reprezentē programmas sintakses struktūru un attiecās uz T-tree datu struktūras, kur katrs valodas AST mezgls reprezentē programmas elementu. AST mezgls var saturēt īpašības (*properties*) un saites ar pārējiem AST mezgliem [28].

Galvenā JetBrains MPS vienība ir projekts, kas sastāv no dažāda veida modeļiem. JetBrains MPS rīkā ir dažāda veida moduļi [34]:

- Risinājums (*Solution*) – dažādu moduļu kopums;
- Valoda (*Language*) – modulis, kurā tiek definēta atkārtoti lietojama valoda, izmantojot dažādus valodas aspektus (sk. 3.2. apakšnodaļu);
- Ģenerators (*Generator*) – moduļa veids, kurā tiek definēti valodas transformācijas likumi citā valodā;
- DevKit – modulis, kas ļauj apvienot dažādas valodas vienā vienībā.

3.2. DSL definēšanas modulis

Lai spētu izveidot DSL MPS rīkā, ir nepieciešams zināt kā JetBrains MPS rīkā var definēt valodu. MPS satur valodas definēšanas moduli, kurā tiek definēti valodas aspekti (*language aspects*). Piemēram, MPS valodas definēšanas modulis satur aspektus, kas ir aprakstīti 3.1.1. – 3.1.6. apakšnodaļās.

3.1.1. Struktūra (Structure)

Lai izveidotu DSL JetBrains MPS rīkā, ir jāapraksta valodas abstraktā sintakse [32]. Valodas abstraktā sintakse JetBrains MPS rīkā tiek definēta, izmantojot valodas struktūras aspektu. Valodas struktūra ir tāds valodas aspekts, kurš ļauj definēt valodas elementus un tos attiecības (*Relationships*) [35].

Lai definētu valodas struktūru, ir jādefinē valodas koncepti (*Concept*). Katrs MPS koncepts ir valodas pamatelements, kas nosaka valodas AST mezgla veidu. Koncepts ir līdzīgs “ķieģelim”, no kura tiek veidota valoda. Koncepts parādās valodas struktūras modelī uzreiz pēc tā izveidošanas [33]. Tāpat MPS koncepts ir līdzīgs klasei objektorientētā programmēšanā – konceptam var norādīt īpašības (*Properties*) un tās bērnus (saites uz pārējiem konceptiem). Konceptu var mantot no cita koncepta, lai mantotu nepieciešamās īpašības. Konceptam var definēt citus aspektus, piemēram, uzvedību redaktorā vai uzvedību koda ģenerēšanas laikā [35].

3.1.2. Redaktors (Editor)

MPS rīkā ir iespējams izveidot valodas redaktora (*Editor*) aspektu. MPS redaktors ir grafiskais vai tekstuālais rīks, kurā var izmantot izveidoto valodu. MPS redaktors ir projekciju redaktors (*Projectional editor*) [28]. Redaktora izmantošanas laikā lietotājs tieši manipulē ar valodas AST - izveido, rediģē AST mezglus. MPS konkrētā sintakse var sastāvēt ne tikai no teksta, bet arī no simboliem (piem., matemātiskiem simboliem), tabulām, grafiskiem elementiem (piem., diagrammām) un formām [36].

Tā kā MPS rīkā tiek izmantots AST, redaktors var paredzēt konstrukcijas, kuras tiks izmantotas tālāk. Redaktors var sniegt ātrus koda labojumus un paredzēt lietotāja nākamās darbības. Kad lietotājs maina kodu, piemēram, izmaina mainīgo nosaukumu vai bloku, valodas AST tiek automātiski pārveidots. Tādējādi MPS redaktori nodrošina dažādas pārstrukturēšanas darbības. Pateicoties AST redaktors var izmantot reāllaika kļūdu pārbaudes mehānismu valodas izmantošanas laikā. Mehānisms pārbauda ievadīto kodu, meklē kļūdas tipus un konstrukcijās. Visas konstatētas kļūdas redaktors izceļ un parāda kļūdu ziņojumus [36].

3.1.3. Ierobežojumi (Constraints)

MPS ierobežojumi nosaka ierobežojumus modelī. Tajā skaitā MPS ierobežojumi apraksta attiecības starp valodas mežgliem un valodas uzvedību. Ierobežojumi var tikt izmantoti, lai ieviestu likumsakarību pārbaudi, objektu atribūtu, tipu un saišu pārbaudi. MPS ierobežojumi tiek izmantoti automātiski valodas modeļu izmantošanas laikā. MPS ierobežojumi ir derīgi kļūdu noteikšanā un validāciju sniegšanā. Kā arī ierobežojumi var tikt definēti, lai testētu valodas modeļus [37].

3.1.4. Uzvedība (Behavior)

Valodas uzvedība apraksta, kā sistēmai jāreaģē un jāapstrādā izstrādātā valoda. Tas nosaka, kā sistēmai jārikojas dažādu darbību izpildes laikā. Tāpat MPS uzvedības aspekts apraksta, kā operācijas savstarpēji mijiedarbojas [38].

3.1.5. Ģenerators (Generator)

JetBrains MPS rīkā eksistē divi AST transformācijas veidi - model-to-model (M2M) un model-to-text (M2T) transformācija. Izmantojot ģeneratora aspektu, tiek definēta M2M transformācija [39]. Šīs transformācijas rezultātā DSL modelis transformējas citā valodas modelī. Izstrādātājs definē nepieciešamos likumus un transformācijas katram AST mezglā tipam, lai definētu kodu, kas tiks uzģenerēts pēc DSL izpildes. Savukārt, izpildot DSL, MPS ģenerators, pielieto definētos likumus un transformācijas, lai uzģenerētu izpildāmo kodu atbilstoši definētiem likumiem [32] [33].

Ģeneratoru izmanto, piemēram, lai:

- transformētu modeļus teksta avota failos (*text source files*), kas pēc tām var tikt kompilēti GPL valodas kompilatorā (Java, C);
- transformētu modeļus teksta dokumentos, piemēram, XML, PDF, LaTeX;
- tieši interpretētu modeļus [32].

3.1.6. TextGen

TextGen aspekts nodrošina M2T transformāciju. M2T transformācijas rezultātā valodas AST mežgli tiek pārveidoti citas GPL izpildāmā kodā, piemēram, Java, C#, JavaScript vai XML. Pēc koda ģenerācijas, uzģenerētais kods var tikt integrēts, kompilēts un izpildīts citā programmā. DSL izveides laikā nav iespējams definēt gan ģeneratora aspektu gan TextGen. Izstrādātājs izvēlās, kādu transformācijas veidu pielietot – M2M vai M2T transformāciju [33].

4. DSL MOODLE TESTU IZVEIDOŠANAI

Šobrīd Moodle platforma (sk. 1. nodaļu) ir populārākā mācību pārvaldības sistēma gan Latvijā, gan visā pasaulē. Moodle platforma satur daudz dažādu aktivitāšu, kas paredzētas efektīvam mācību procesam. Viena no svarīgākajām aktivitātēm Moodle platformā ir interaktīvie testi. Gandrīz visosursos pasniedzēji veido interaktīvus testus, lai pārbaudītu studentu zināšanas. Tomēr interaktīvo testu veidošanas procesu var uzlabot. Lai pilnveidotu šo procesu, bakalaura darba ietvaros tika izstrādāts:

- domēna specifiskā valoda Moodle testu izveidošanai;
- Moodle spraudnis.

Domēna specifiskās valodas izveidei tika izvēlēts JetBrains MPS rīks (sk. 3. nodaļu). JetBrains MPS tika izvēlēts vairāku iemeslu dēļ:

- šajā rīkā ir iespējams izstrādāt tekstuālo domēna specifisko valodu, kas visvairāk piemērota izstrādājama valodai;
- šis rīks nodrošina vienkāršāku lietotāja saskarni DSL izstrādei, kas atvieglo DSL veidošanu.

Šī nodaļa satur DSL domēna aprakstu (sk. 4.1. apakšnodaļu), DSL sintakses aprakstu (sk. 4.2. apakšnodaļu), DSL definēšanas moduļa aprakstu (sk. 4.3. apakšnodaļu), Moodle spraudņa veidošanas aprakstu (sk. 4.3. apakšnodaļu) un DSL izmantošanas scenāriju, kas ir aprakstīts 4.4. apakšnodaļā.

4.1. DSL domēns

Tika izstrādāta domēna specifiskā valoda Moodle testiem. Lai precīzāk parādītu jaunas valodas iespējas, apskatīsim problēmu uz piemēra.

Pieņemsim, ka pasniedzējs pieslēgts Moodle kursam. Lai izveidotu testu un pievienotu jautājumus testam, pasniedzējs var pāriet no Moodle kursa izvēlnes uz kursa jautājumu bankas sadaļu. Jautājumu banka jau ir aizpildīta ar jautājumiem un jautājumi tika sadalīti kategorijās (jautājumu bankas lapu var redzēt 4.1. attēlā). Lai izveidotu jaunu testu, ir jāatlasa nepieciešamie jautājumi, kas jāiekļauj testā. Pēc uzdevumu pievienošanas ir nepieciešams piemērot dažādus testa iestatījumus, piemēram: ierobežot testa pieejamības laiku, testa atvēršanas laiku, jautājumu jaukšanu, ierobežot piekļuvi testam utt.

Ja ir nepieciešams izveidot vairākus testa variantus, jāizveido vairāki testi. Līdz ar testu izveidošana Moodle platformā var būt darbietilpīgs process, pat ja jautājumi jau ir izveidoti un

sadalīti kategorijās. Viens no problēmas risinājumiem ir izstrādāt domēna specifisko valodu testu izveidošanai, kas atvieglo un paātrina Moodle testu izveidošanas procesu.

LMS Moodle Home Dashboard My courses Site administration

Question bank

Select a category: Eksamens (15) ▾

Eksāmena jautājumi

No tag filters applied

Filter by tags... ▾

☐ Show question text in the question list

[Search options](#) ▾

☐ Also show questions from subcategories

☒ Also show old questions

Create a new question ...

	Question	Actions	Status	Version	Created by	Comments
	Question name / ID number				First name / Last name / Date	
<input type="checkbox"/>	LINQ lazy loading ✎ writable	Edit ▾	Ready ▾	v2	Admin User 10 April 2023, 5:41 PM	0
<input type="checkbox"/>	LINQ Take ✎ writable	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 5:38 PM	0
<input type="checkbox"/>	C# general ✎	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 2:59 PM	0
<input type="checkbox"/>	EF general ✎	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 2:44 PM	0
<input type="checkbox"/>	EF general ✎	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 2:49 PM	0
<input type="checkbox"/>	EF querying data ✎	Edit ▾	Ready ▾	v2	Admin User 10 April 2023, 2:14 PM	0
<input type="checkbox"/>	EF querying data ✎	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 2:32 PM	0
<input type="checkbox"/>	EF querying data ✎	Edit ▾	Ready ▾	v1	Admin User 10 April 2023, 2:34 PM	0

4.1. att. – Moodle kursa jautājumu banka

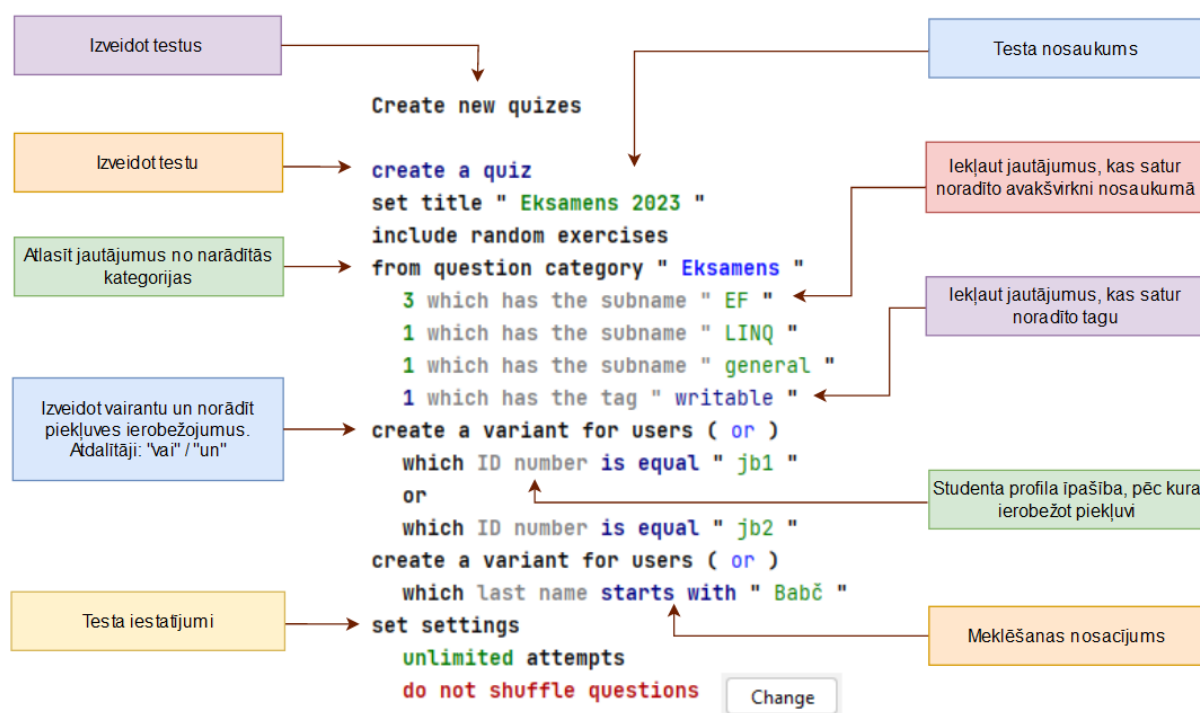
Līdz ar to bakalaura darba ietvaros tika izstrādāta domēna specifiskā valoda Moodle testu izveidošanai. Jaunā domēna specifiskā valoda ļauj:

- izveidot vairākus testus;
- izveidot vairākus testa variantus;
- norādīt testa nosaukumu;
- iekļaut testā nejaušus uzdevumus no kursa jautājumu bankas un norādītās jautājumu kategorijas pēc nosaukuma apakšvirknes vai taga;
- ierobežot piekļuvi katram testa variantam;
- norādīt testa iestatījumus (mēģinājumu skaits, jautājumu jaukšana).

DSL sintakse ir detalizēta 4.2. apakšnodaļā.

4.2. DSL sintakse

Apskatīsim DSL sintaksi, izmantojot 4.2. attēlā parādīto piemēru.



4.2. att. – DSL skripts Moodle testa izveidošanai

DSL skripts sākas ar rindu *“Create new quizzes”*, kas apzīmē, ka tālāk tiks definēti testi. Nākamā rinda *“create a quiz”* apzīmē turpmāku testa definēšanu.

Pēc tam seko *“include random exercises”* rinda, kas nozīmē turpmāku jautājumu atlases definēšanu. Šajā blokā lietotājs norāda jautājumus, kas tiks nejauši izvēlēti testa izveidošanas laikā. Jautājumi tiek atlasīti, ņemot vērā jautājumu kategoriju, kuru norāda lietotājs tekstuālā veidā *“from question category”* rindā. Domēna specifiskā valoda ļauj atlasīt jautājumus pēc apakšvirknes vai taga. Katrai norādītajai nosaukuma apakšvirknei un tagam ir jānorāda cik daudz atbilstošu jautājumu no kursa jautājumu bankas ir jāatlasa un jāiekļauj testā (var būt no 1 līdz 100).

Pēc jautājumu definēšanas bloka, seko testa variantu definēšanas bloks. Pēc noklusējuma, katram testam ir jānorāda vismaz viens variants. Pievienot variantu ir iespējams, izmantojot valodas rindu *“create a variant for users”*. Katram testa variantam var norādīt piekļuves ierobežojumus. Katrā piekļuves ierobežojumā ir jānorāda profila īpašība. Tas var būt:

- lietotājevārds (*username*);
- vārds (*first name*);
- uzvārds (*last name*).

Tāpat ir jāieraksta meklēšanas nosacījums. Meklēšanas nosacījums var būt viens no sekojošiem elementiem:

- vienāds (*is equal*);
- satur (*contains*);
- sākas ar (*starts with*).

Ir iespējams pievienot vairākus piekļuves ierobežojumus vienam testa variantam. Piekļuves ierobežojumus var kombinēt, izmantojot “vai (*or*)” / “un (*and*)” atdalītāju. Piemēram, pēc skripta izpildes, kas ir redzams 4.2. attēlā, tiks izveidoti divi testa varianti, kuros piekļuves ierobežojumi atšķiras. Pirmais testa variants būs pieejams lietotājiem ar identifikatoru "jb1" vai "jb2". Otrais testa variants būs pieejams lietotājiem, kuru uzvārds sākas ar virkni "Babč". Pēc noklusējuma tiek rādīta “*all course users*” rinda, kura nozīmē, ka testa variantam var piekļaut visi kursa lietotāji. Visi izveidotie testi pēc DSL skripta izpildes būs paslēpti no studentiem un visus piekļuves ierobežotājus būs iespējams pārbaudīt Moodle testa iestatījumos. Lai testi būtu pieejami studentiem, Moodle testa iestatījumos ir jāatver testa aktivitāte vai jāiestata testa atvēršanas laiks.

Domēna specifiskās valodas beigās ir testa iestatījumu bloks. Iestatījumu blokā ir iespējams norādīt:

- mēģinājumu skaitu, kas var būt bezlimita (*unlimited*) vai 1-10 diapazonā;
- jautājumu jaukšanu (jā/nē).

Var secināt, ka izstrādātā domēna specifiskā valoda līdzīga valodai, ko mēs lietojam ikdienā. Tā atgādina veidu, kā mēs aprakstām vēlamu rezultātu no programmas, izmantojot cilvēku valodu. Tādēļ valodas sintakse ir viegli uztverama un saprotama. Līdz ar to jauno domēna specifisko valodu var izmantot lietotāji bez iepriekšējām programmēšanas prasmēm.

4.3. DSL definēšanas modulis

DSL izstrādei tika izmantots JetBrains MPS rīks (sk. 3. nodaļu). Lai izveidotu DSL, JetBrains rīkā tika izveidots valodas definēšanas modulis, kas satur dažādus valodas aspektus, piemēram: struktūru, redaktoru, ierobežojumus, ģeneratoru utt. (sk. 3.1.1. – 3.1.6. apakšnodaļu). Šajā nodaļā ir aprakstīti nozīmīgākie izstrādātās DSL definēšanas moduļa aspekti: struktūra, redaktors un ģenerators (sk. 4.2.1. – 4.2.3. apakšnodaļu). Šie aspekti apraksta izstrādātās domēna specifiskās valodas sintaksi un semantiku.

4.3.1. DSL struktūras aspekts

Vispirms JetBrains MPS rīkā tika definēta valodas struktūra (sk. 3.1.1. apakšnodaļu), kas satur:

- valodas konceptus (*Concepts*);
- datu tipus (*Constrained data type*);
- sarakstu tipus (*Enumeration*).

Katram konceptam tika definēts:

- nosaukums;
- īpašības (*Properties*);
- attiecības ar citiem konceptiem (*Relationships*).

Tika izveidota DSL struktūras klašu diagramma, kas ir redzama 4.3. attēlā. Apskatīsim DSL struktūras klašu diagrammu detalizētāk. DSL saknes koncepts ir QuizesBlock, kas apzīmē bloku ar Moodle testiem. Katrs QuizesBlock koncepts satur vismaz vienu Quiz konceptu (QuizesBlock koncepta bērns). Savukārt katrs Quiz sastāv no:

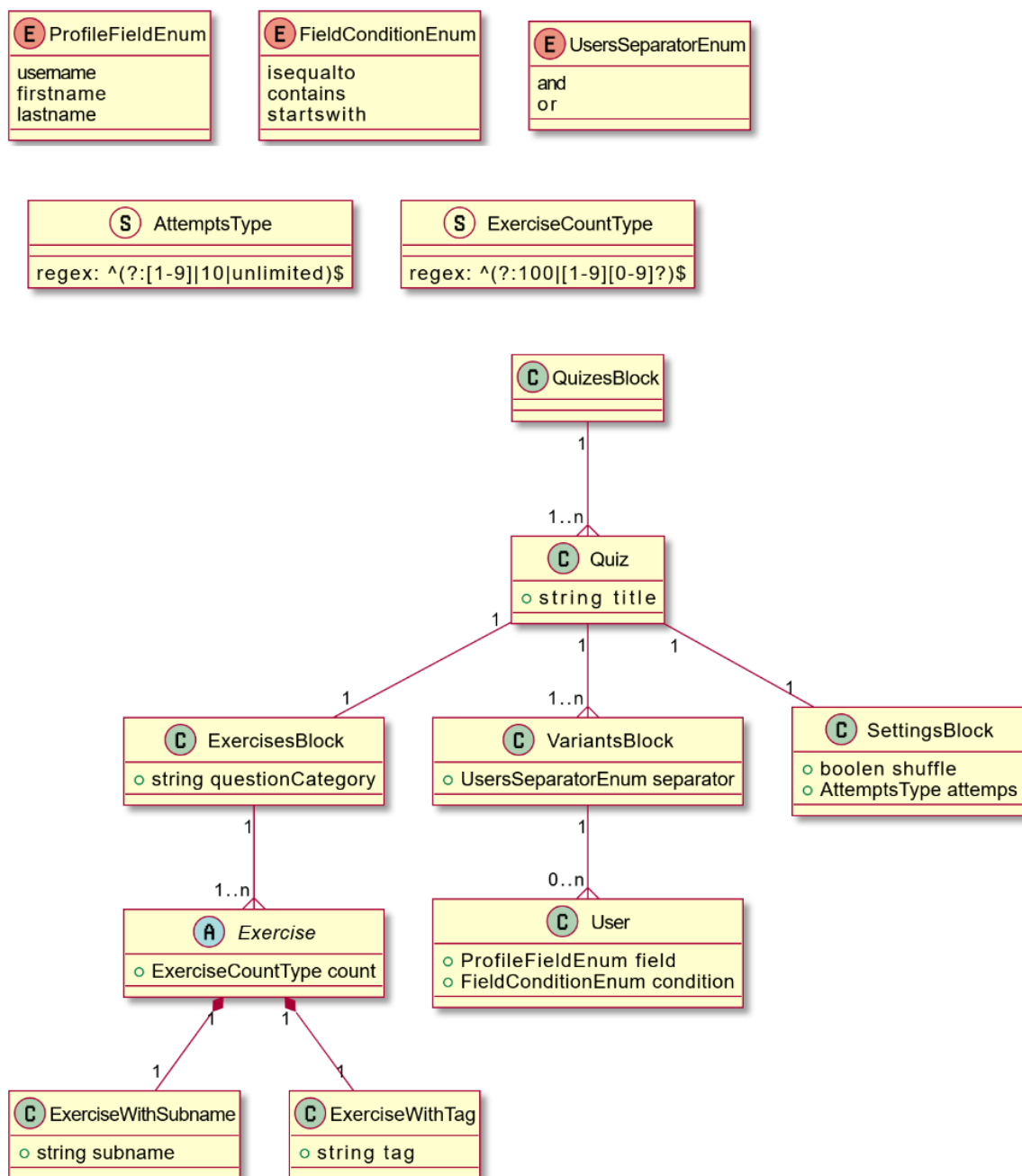
- uzdevumu bloka (*ExercisesBlock*);
- vismaz viena testa varianta bloka (*VariantsBlock*);
- iestatījumu bloka (*SettingsBlock*).

Uzdevumu (*ExercisesBlock*) bloks satur vismaz vienu bērnu – bērnu, kas tiek mantots no abstraktas klases *Exercise*. Tas var būt uzdevums, kas satur apakšvirkni nosaukumā (*ExerciseWithSubname*) vai uzdevums, kas satur tagu (*ExerciseWithTag*). Katrs uzdevums satur īpašību “count” ar tipu “*ExerciseCountType*”. “*ExerciseCountType*” apzīmē skaitlisko vērtību no 1 līdz 100. Šis tips tika definēts, izmantojot regulāro izteiksmi $^(?:100|[1-9][0-9]?)\$$.

Testa variantu bloks (*VariantsBlock*) ir bloks, kas apzīmē testa variantu un piekļuves ierobežojumus tam (*User*). Piekļuves ierobežojumus var atdalīt, izmantojot atdalītāju. Atdalītājs saglabājas User koncepta “separator” īpašībā. Kā jau tika minēts 4.2. apakšnodaļā, atdalītājs var būt “or” vai “and”. Tāpat katrs piekļuves ierobežojuma mezgls satur īpašību “field”, kas apzīmē lietotāja profila īpašību pēc kura ierobežot piekļuvi un īpašību “condition”, kas apzīmē meklēšanas nosacījuma vērtību.

Iestatījumu bloks (*SettingsBlock*) ir bloks, kurā lietotājs definē testa iestatījumus: jautājumu jaukšanu un mēģinājumu skaitu. Mēģinājumu skaits ir ar tipu *AttemptsType*, kas tika definēts, izmantojot regulāro izteiksmi $^(?:[1-9]|10|unlimited)\$$. Tas nozīmē, ka mēģinājumu skaits var būt neierobežots vai no 1 līdz 10.

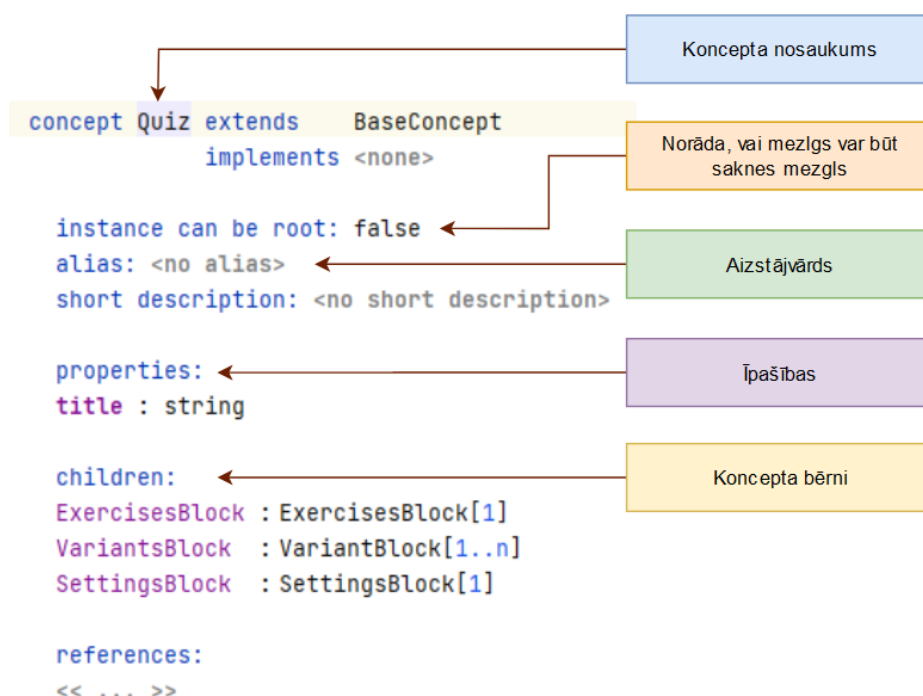
Moodle Quiz - Class Diagram



4.3. att. – DSL struktūras klašu diagramma

DSL struktūras klašu diagrammā var redzēt, ka DSL struktūrā pievienoti ne tikai koncepti, bet arī datu tipi (*Constrained data type*) un sarakstu tipi (*Enumeration*). Tie tiek izmantoti konceptos, lai norādītu īpašību tipus.

Apskatīsim Quiz koncepta definējumu JetBrains MPS rīkā (sk. 4.4. attēlu). Vispirms jāieraksta koncepta nosaukums. Tālāk seko koncepta iestatījums, kas apzīmē, vai koncepts var būt AST saknes mezgls vai nē. Izstrādātās DSL Quiz koncepts nevar būt AST saknes mezgls, līdz ar to iestatījums “*instance can be root*” ir “*false*”. Pēc tam konceptam var norādīt aizstājvārdu un īso aprakstu. Bet tā kā Quiz koncepts netiks izmantots DSL redaktora pabeigšanas laukos (*completion boxes*), tam netiek norādīts aizstājvārds un īsais apraksts. Quiz koncepts satur 1 (vienu) īpašību (*property*) – testa nosaukumu “*title*”. Tāpat Quiz koncepts satur 3 (trīs) bērnu konceptus (*children*): uzdevumu bloku (*ExercisesBlock*), testa variantu blokus (*VariantsBlock*) un iestatījumu bloku (*SettingsBlock*).



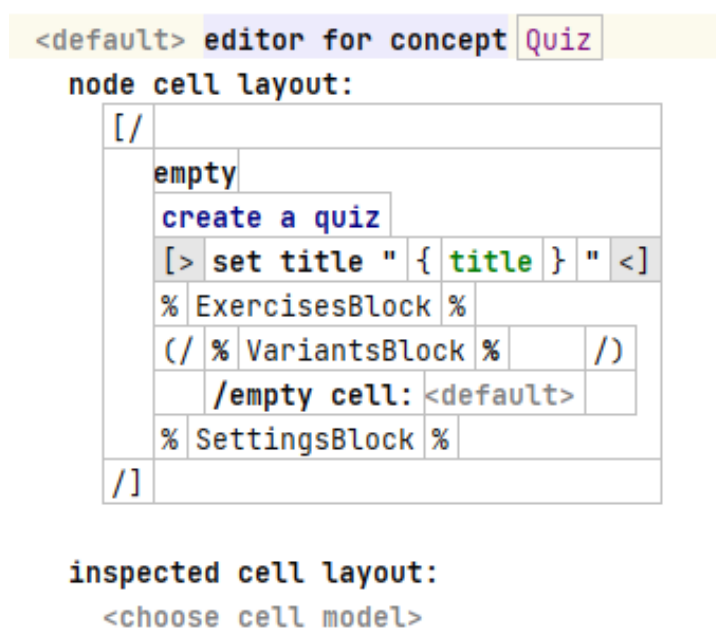
4.4. att. – DSL Quiz koncepts JetBrains MPS rīkā

Tā kā DSL struktūras definēšanai tika izveidoti vairāki koncepti, tika izveidota tabula (sk. 1. pielikumu), kurā var redzēt visus DSL definētus konceptus.

Var secināt, ka DSL struktūras aspekts nosaka valodas sintakses likumus. Turpmāk, pamatojoties uz sintakses likumiem, valodas elementus var izveidot un pārvaldīt MPS vidē. Tomēr, lai būtu iespējams manipulēt ar valodas elementiem, ir jādefinē DSL redaktora aspekts (sk. 3.1.2. apakšnodaļu). Līdz ar to nākamajā apakšnodaļā (sk. 4.3.2. apakšnodaļu) tiks definēts DSL redaktora aspekts.

4.3.2. DSL redaktora aspekts

Tā kā bakalaura darba ietvaros tiek izstrādāts ārējais DSL, ir jāizveido mehānisms, ar kuru būs iespējams izveidot un rediģēt valodas AST. Valodas redaktora definēšana JetBrains MPS rīkā ļauj izveidot mehānismu, ar kuru ir iespējams izmantot DSL sintakses elementus ērti un intuitīvi. Izstrādātās DSL sintakses elementi ir aprakstīti 4.3.1. apakšnodaļā. Katram DSL konceptam tika definēta konkrētā sintakse, kas nosaka kā katrs koncepts izskatīsies redaktorā. Piemēram, 4.5. attēlā var redzēt Quiz koncepta redaktoru.

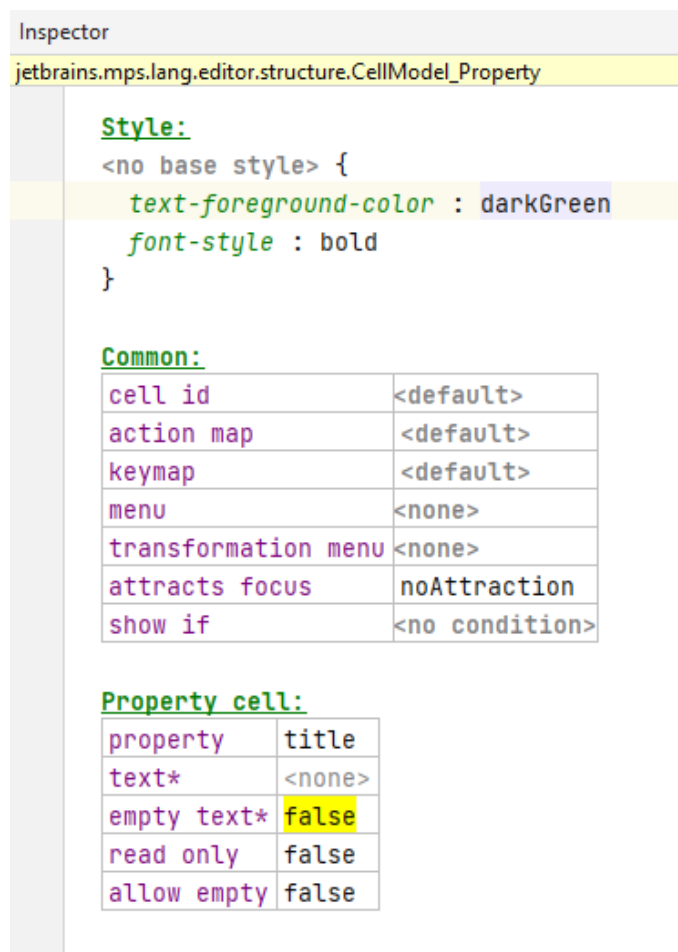


4.5. att. – Quiz koncepta redaktors

Quiz koncepta redaktors satur:

- tukšu rindu;
- “new quiz” rindu turpmākai testa definēšanai;
- horizontālo kolekciju, kurā ir “set title” vārdu virkne un testa nosaukuma īpašības vērtība, kuru lietotājs var rediģēt;
- saite uz uzdevumu bloku (*ExercisesBlock*);
- saite uz testa variantu bloku (*VariantsBlock*);
- saite uz iestatījumu bloku (*SettingsBlock*).

Tāpat koncepta redaktora elementiem tiek iestatīts izskats redaktorā. Piemēram, vārds “*new quiz*” ir zilā krāsā, bet testa nosaukuma īpašības vērtība zaļā krāsā. Katram mezglam to var iestatīt mezgla inspektorā. Piemēram, 4.6. attēlā var redzēt Quiz mezgla inspektoru - īpašības “*title*” vērtības izskata definējumu.



4.6. att. – Quiz mezgla inspektors – nosaukuma īpašība

Līdzīgā veidā tika izveidoti pārējie DSL konceptu redaktori. Visus DSL konceptu redaktorus var redzēt DSL konceptu redaktoru tabulā (sk. 2. pielikumu).

4.3.3. DSL ģeneratora aspekts

4.2.1. un 4.2.2. apakšnodaļā tika aprakstīta DSL valodas sintakse – valodas elementi, to attiecības un redaktors. Tomēr, lai pilnībā definētu DSL, ir jādefinē valodas semantika. Semantika piešķir valodai nozīmi un nosaka tās uzvedību. JetBrains MPS rīkā var aprakstīt valodas semantiku, izmantojot ierobežojumus (sk. 3.1.3. aspektu), uzvedību (3.1.4. aspektu), ģeneratoru (sk. 3.1.5. apakšnodaļu) un citus aspektus. Šajā apakšnodaļā ir aprakstīts DSL ģeneratora aspekts.

Tika definēta DSL model-to-model transformācija (*model-to-model transformation*) paplašināmās iezīmēšanas valodā (*Extensible Markup Language – XML*). Modeļu transformācijai tika definēta kartēšanas konfigurācija (*mapping configuration*), kas ir redzama 4.8. attēlā. Tā satur saknes kartēšanas konfigurācijas likumus (*root mapping rules*) un reducēšanas likumus (*reduction rules*). Katrs reducēšanas likums satur pārveidošanas veidni, balstoties uz kura, JetBrains MPS rīks automātiski ģenerē XML elementus. Piemēram, 4.7. attēlā var redzēt Quiz koncepta pārveidošanas veidni.

```
template reduce_Quiz
input Quiz

parameters
<< ... >>

content node:
<TF [ <moodleTest>
    <title>${ } </title>
    ${COPY_SRC}${ } ${COPY_SRC}${ } ${COPY_SRC}${ }
</moodleTest> ] TF>
```

4.7. att. – Quiz koncepta pārveidošanas veidne

Quiz koncepta pārveidošanas veidnē var redzēt, ka katram Quiz koncepta mezglam tiek ģenerēts <moodleTest> elements. Šī taga iekšā tiek ģenerēts:

- XML elements <title>, kas nosaka testa nosaukuma vērtību;
- bērnu mezglu pārveidošanu XML formātā (uzdevumu, variantu un iestatījumu bloku redukcijas likumu piemērošana).

Tika izveidota tabula (sk. 3. pielikumu) kurā parādīti visi DSL konceptu redukcijas likumi.

root mapping rules:

```
[concept      QuizzesBlock ] --> map_QuizzesBlock
[inheritors   false        ]
[condition    <always>     ]
[keep input root default  ]
```

weaving rules:

<< ... >>

reduction rules:

```
[concept      Quiz          ] --> reduce_Quiz
[inheritors   false        ]
[condition    <always>     ]
```

```
[concept      ExercisesBlock ] --> reduce_ExercisesBlock
[inheritors   false        ]
[condition    <always>     ]
```

```
[concept      ExerciseWithSubname ] --> reduce_ExerciseWithSubname
[inheritors   false        ]
[condition    <always>     ]
```

```
[concept      ExerciseWithTag ] --> reduce_ExerciseWithTag
[inheritors   false        ]
[condition    <always>     ]
```

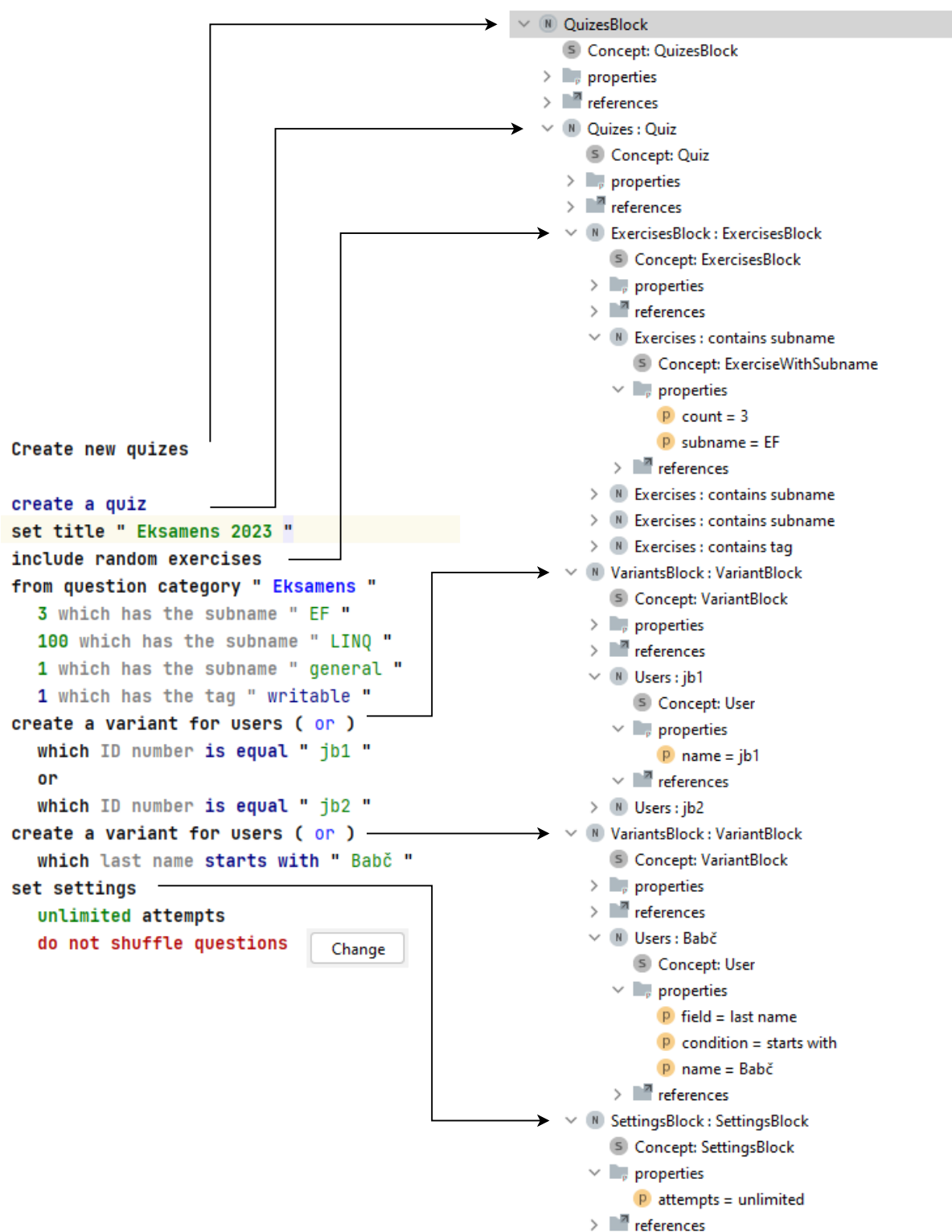
```
[concept      SettingsBlock ] --> reduce_SettingsBlock
[inheritors   false        ]
[condition    <always>     ]
```

```
[concept      VariantBlock ] --> reduce_VariantBlock
[inheritors   false        ]
[condition    <always>     ]
```

```
[concept      User          ] --> reduce_User
[inheritors   false        ]
[condition    <always>     ]
```

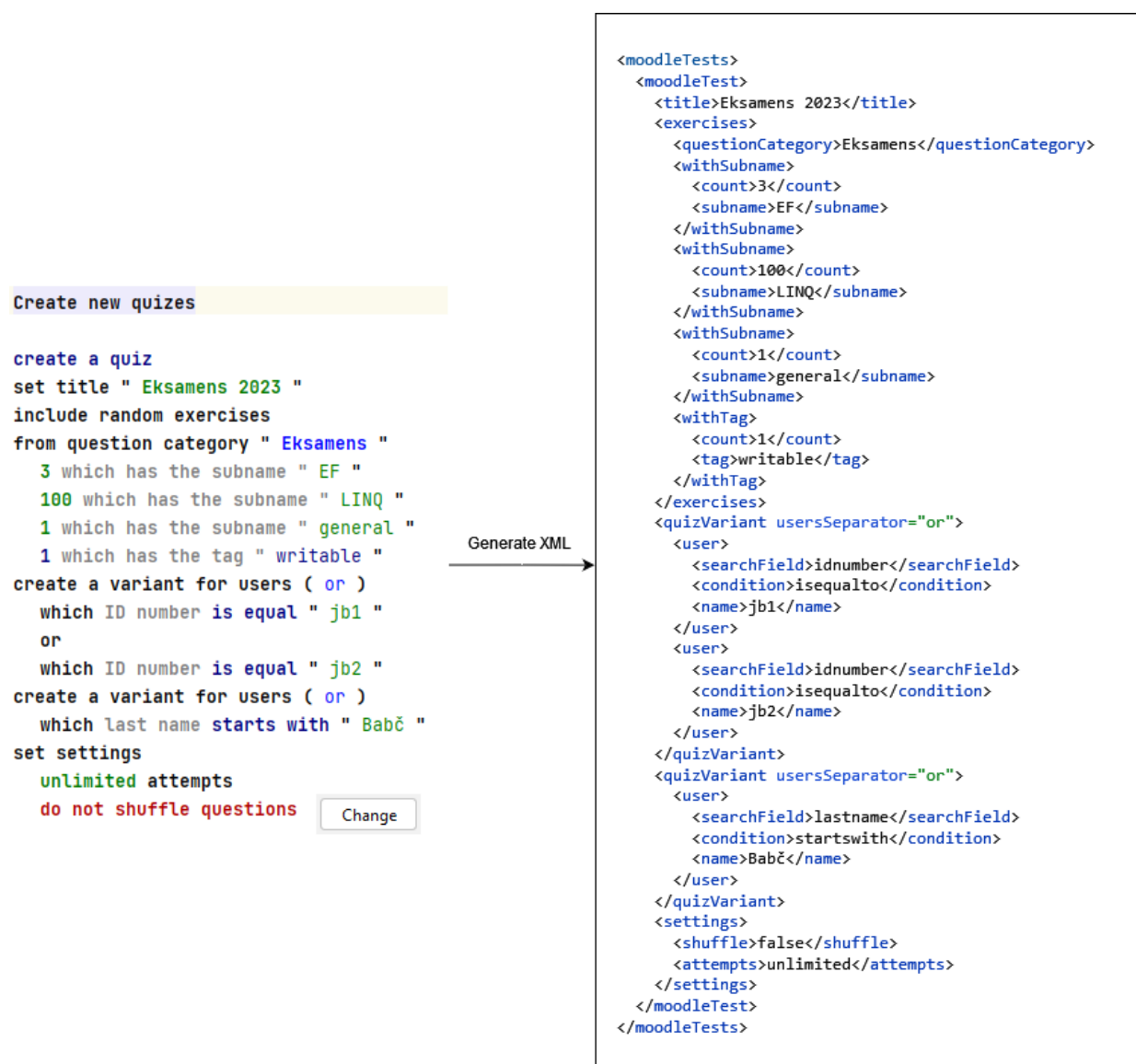
4.8. att. – DSL ģenerators kartēšanas konfigurācija

JetBrains MPS glabā ievadīto kodu abstraktā sintakses kokā. Tieši tāpēc ģenerators saprot kādu ģeneratora veidni izmantot. Piemēram, 4.9. attēlā var redzēt abstrakto sintakses koku, kuru JetBrains MPS rīks glabā atbilstoši ievadītajam kodam.



4.9. att. – DSL skripta abstraktais sintakses koks

XML failu ir iespējams uzģenerēt DSL redaktorā, noklikšķinot uz pogas “*Preview generated text*”. Uzģenerēto XML faila saturu atbilstoši DSL kodam var redzēt 4.10. attēlā. Acīmredzami, ka saprast DSL kodu ir vienkāršāk, nekā XML iezīmēšanas valodu.



4.10. att. – XML faila ģenerēšana

4.4. Moodle spraudnis

Lai būtu iespējams apstrādāt XML failu un izveidot testus Moodle platformā, tika izveidots Moodle spraudnis (sk. 1.1. nodaļu). Jaunais Moodle spraudnis satur:

- formu, kurā lietotājs var augšupielādēt XML failu Moodle platformā;
- saiti kursa izvēlnē kursa pasniedzējiem uz XML faila augšupielādes formu;
- funkcionalitāti testu izveidošanai (XML faila parsēšana un apstrāde).

Būtībā jaunajā spraudnī tika realizēts DSL interpretators. Jaunais Moodle spraudnis tika izstrādāts ņemot vērā Moodle arhitektūru, kas tika aprakstīta 1.1. apakšnodaļā, PHP programmēšanas valodā.

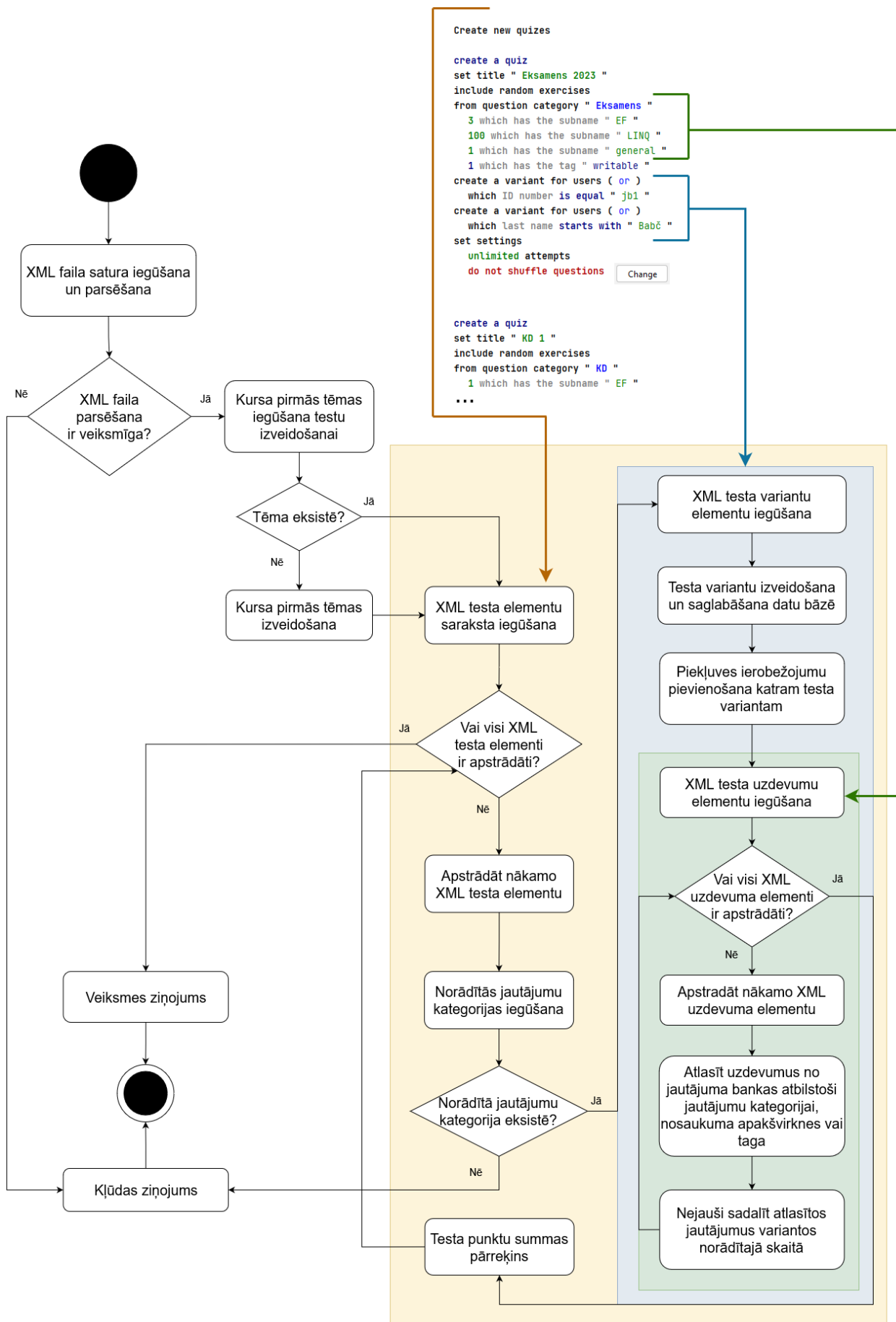
4.4.1. Spraudņa izveidošanas process

Vispirms, tika lejupielādēts phpMyAdmin bezmaksas programmatūras rīks. Šis rīks satur MySQL datu bāzes pārvaldības sistēmu. Tika lejupielādēts Moodle pirmkods, kas ir pieejams publiskajā repozitorijā [5]. Moodle lejupielādes laikā izveidojās tukša datu bāze MySQL pārvaldības sistēmā un visas nepieciešamas tabulas Moodle veiksmīgai darbībai. Ņemot vērā Moodle arhitektūru, visi lokālie spraudņi ir jāizstrādā Moodle pirmkoda “local” mapē. Tādēļ “local” mapē tika izveidota spraudņa mape. “moodletests” Spraudņa failos, kuros ir nepieciešams izmantot datubāzi, tika pieslēgts “config.php” fails, kurā tiek glabāta datu bāzes pieslēgšanas konfigurācija (sk. 1.2. apakšnodaļu).

Tika izveidota forma, kurā ir iespējams augšupielādēt XML failu Moodle testu izveidošanai. Izveidoto formu var redzēt 9. pielikumā. Pēc jaunās formas izveides, jaunajā spraudnī tika realizēts DSL interpretators. DSL interpretators izveido Moodle testus un tos variantus, balstoties uz XML dokumenta saturu. DSL interpretatora algoritms tiks detalizēts 4.4.2. apakšnodaļā.

4.4.2. DSL interpretatora algoritms

Moodle testu izveidošanai tika izstrādāts DSL interpretators jaunajā Moodle spraudnī. Lai parādītu testu izveidošanas algoritmu, tika izveidota diagramma, kas ir redzama 4.11. attēlā. Apskatīsim 4.11. attēlā redzamo Moodle testu izveidošanas algoritmu detalizētāk. Sākumā notiek XML dokumenta parsēšana un apstrāde. XML parsēšanas laikā notiek faila lasīšana, tā struktūras validēšana un pārveidošana nepieciešamajā objektā turpmākai testu izveidošanai. Ja XML parsēšanas laikā notika kļūda, attiecīgi tiks attēlots kļūdas ziņojums. Ja XML parsēšana ir veiksmīga, notiek testu izveidošana atbilstoši augšupielādētajam XML dokumenta saturam.



4.11. att. – Testu izveidošana Moodle spraidnī

Pēc objekta izveidošanas, notiek testu izveidošana sekojošā secībā:

- kursa pirmās tēmas iegūšana, kurā jā saglabā tests;
- katram XML testa elementam:
 - norādītās jautājumu kategorijas iegūšana;
 - tiek veidoti un saglabāti testu varianti datu bāzē ar visiem norādītiem iestatījumiem (kursa modulis, kursa tēma, nosaukums, mēģinājumu skaits, jaukšanas iestatījums utt.);
 - tiek pievienoti norādītie piekļuves ierobežojumi;
 - jautājumu atlase no jautājumu bankas no norādītas jautājumu kategorijas (tiek ņemti vērā norādītie apakšnosaukumi/tagi un uzdevumu skaits);
 - testa punktu skaita pārrēķins.

Pirms DSL interpretatora izveidošanas bakalaura darba autorei nebija pieredzes Moodle platformas izstrādē. Bija nepieciešams izprast Moodle arhitektūru, tās datu bāzes struktūru (sk. 1. nodaļu), lai izveidotu DSL interpretatoru. Tāpat bija nepieciešams izpētīt dokumentāciju un platformas pirmkodu, lai atrastu eksistējošās metodes dažādu darbību veikšanai (piemēram, testa izveidošanai un kešatmiņas atjaunošanai). Koda fragments, kurā tiek realizēta testa variantu izveidošana un to piekļuves ierobežojumu pievienošana, ir redzams 4.12. attēlā. Koda fragmentus, kuros tiek realizēta jautājumu atlasīšana no kursa jautājumu bankas, izmantojot nosaukuma apakšvirkni un tagu, var redzēt 10. un 11. pielikumā.

```
// Create quiz variants
$variants = array();
foreach ($quiz->variants as $quizVariant) {
    // Create new quiz
    $quizId = create_quiz($quiz, $courseId, $module->id, $section->id, $quiz->title, $DB);
    $newQuizVariant = $DB->get_record('quiz', array('id' => $quizId));
    array_push($variants, $newQuizVariant);

    // Add quizzes restrictions
    $restrictions = array();
    foreach($quizVariant->users as $user) {
        array_push($restrictions, \availability_profile\condition::get_json(
            false,
            $user->filter,
            get_user_field_condition($user->condition),
            $user->name));
    }

    if (!empty($restrictions)){
        $coursemodule = $DB->get_record('course_modules', array('instance'=>$quizId));
        $restriction = \core_availability\tree::get_root_json($restrictions, get_users_separator($quizVariant->usersSeparator));
        $DB->set_field(
            'course_modules',
            'availability',
            json_encode($restriction),
            ['id' => $coursemodule->id]);
        rebuild_course_cache($courseId, true);
    }
}
```

4.12. att. – Koda fragments – testa variantu izveidošana un to piekļuves ierobežojumu pievienošana

4.5. DSL izmantošanas piemērs

Šajā nodaļā ir aprakstīts viens izstrādātā DSL testēšanas piemērs. Šis testēšanas piemēra mērķis ir pārbaudīt izstrādātā DSL funkcionalitāti un darbību, kā arī atklāt iespējamus defektus un kļūdas jaunajā Moodle ietvarā. Tāpat parādīt kā ir iespējams izmantot izstrādāto domēna specifisko valodu. DSL testēšanai tika izmantoti Latvijas Universitātes .NET kursa iepriekšējo gadu eksāmena uzdevumu nosaukumi.

Vispirms tika izveidots kurss Moodle sistēmā. Izveidotajā kursā tika pievienota jauna jautājumu kategorija eksāmena jautājumiem. Šajā kategorijā tika pievienoti 49 (četrdesmit deviņi) jautājumi, kuri ir redzami 4. pielikumā. Jautājumu banka sniedz jautājumu sarakstu, no kura tiks nejauši izvēlēti jautājumi atbilstoši nosaukumu apakšvirknēm un tagiem. Lai apzīmētu katra uzdevuma tipu, katram jautājumam tika pievienoti tagi. Katram jautājumam atbilstošo tagu var redzēt 4. pielikuma tabulā “Tags” kolonnā.

Dotajā DSL izmantošanas piemērā ir nepieciešams izveidot 4 (četrus) testa variantus. Katra testa variantā jāiekļauj 11 (vienpadsmit) uzdevumi, kuri ir redzami 4.1. tabulā. Kopā par katru testa variantu var iegūt 20 punktus.

4.1. tabula – Testa struktūra

Tips	Punktu skaits	Skaits	Punkti grupai
MPĪ	2	1	2
Kods	2	2	4
LINQ	2	1	2
Funkcija	2	1	2
PrivatePublic	2	1	2
Teorija	2	1	2
Advanced	2	1	2
Tests	1	2	2
Advanced	2	1	2
		11	20

Testu izveidošanai tika izmantota izstrādātā domēna specifiskā valoda. DSL skripta izveidošanai tika izmantots JetBrains MPS rīks. Pēc JetBrains MPS atvēršanas, ir jāievada kods izveidotā DSL valodā MPS rīkā. Šo informāciju lietotājs ievada strukturētā veidā, ievērojot jaunās DSL struktūru. MPS redaktors palīdz lietotājam ātri rakstīt kodu, izmantojot priekšsaizpildīšanu. Nepieciešamās izvēles gadījumā ir jānospiež CTRL+ALT pogu kombināciju, lai redzētu redaktora piedāvājumus (sk. 4.13. attēlu).

Create new quizzes

```
create a quiz
set title " Eksamens 2023 "
include random exercises
from question category " Eksamens "
  1 which has the tag " MPĪ "
  2 which has the tag " Kods "
  1 which has the tag " LINQ "
  1 which has the tag " Funkcija "
  1 which has the tag " PrivatePublic "
  1 which has the tag " Teorija "
  1 which has the tag " Advanced "
  2 which has the tag " Tests "
  1 which has the tag " Advanced "
create a variant for users ( or )
  which ID number  " jjb1 "
create a variant for users ( or )
  which ID number ☐ contains
create a variant for users ( or )
  which ID number ☐ is equal
create a variant for users ( or )
  which ID number ☐ starts with
create a variant for users ( or )
  which ID number is equal " jjb4 "
set settings
  1 attempts
  do not shuffle questions
```

Change

Create new quizzes

```
create a quiz
set title " Eksamens 2023 "
include random exercises
from question category " Eksamens "
  1 which has the tag " MPĪ "
  2 which has the tag " Kods "
  1 which has the tag " LINQ "
  1 which has the tag " Funkcija "
  1 which has the tag " PrivatePublic "
  1 which has the tag " Teorija "
  1 which has the tag " Advanced "
  2 which has the tag " Tests "
  1 which has the tag " Advanced "
create a variant for users ( or )
  which ID number is equal " jjb1 "
create a variant for users ( or )
  which ID number is equal " jjb2 "
create a variant for users ( or )
  which ID number is equal " jjb3 "
create a variant for users ( or )
  which ID number is equal " jjb4 "
set settings
  1 attempts
  do not shuffle questions
```

Change

4.13. att. – JetBrains MPS redaktora piedāvājumi

4.14. att. – DSL skripts Moodle testu izveidošanai

Izveidoto DSL skriptu var redzēt 4.14. attēlā. DSL skriptā tika definēts viens tests ar nosaukumu “Eksamens 2023”. Pēc tām tika norādīti uzdevumi, kuri jāiekļauj testa variantos. Tika norādīta jaunā jautājumu kategorija “Eksamens”, kurā tiek glabāti visi izveidotie eksāmena uzdevumi. No norādītās jautājumu kategorijas tiks nejauši izvēlēti jautājumi testa izveidošanas laikā. Tika norādīti 11 (vienpadsmit) jautājumi, kuri jāiekļauj testa variantos. Dotajā piemērā jautājumu atlase notiek balstoties uz norādītajiem tagiem. Tāpat tika izveidoti 4 (četri) testa varianti. Katram variantam tika definēti piekļuves ierobežojumi. Izveidotie testa varianti būs pieejami studentiem ar “jb1”, “jb2”, “jb3” un “jb4” identifikatoru. Tāpat skripta beigās tika norādīti iestatījumi: ir atļauts tikai 1 (viens) mēģinājums un secīga jautājumu kārtība testā.

Kad testu informācija ir aizpildīta, ir jāģenerē XML dokuments. Nospiežot pogu “*Preview generated text*” JetBrains MPS redaktorā, tiks ģenerēts XML dokuments, balstoties uz ievadīto kodu un DSL transformēšanas likumiem. Uzģenerētais XML ir jāsavienā xml paplašinājumā. Uzģenerētā XML dokumenta saturu var redzēt 5. pielikumā.

Pēc DSL skripta izveidošanas un XML faila ģenerēšanas, saglabātais fails ir jāaugšupielādē jaunajā XML augšupielādes formā. Izstrādāto XML augšupielādes formu var redzēt 9. pielikumā. Lai uzģenerētu testus, jānoklikšķina uz pogas “Izveidot”. Pēc XML faila augšupielādes notiek XML faila satura pārbaude. Ja XML faila validēšanas laikā notika kļūda, attiecīgi attēlots kļūdas ziņojums. Ja XML tika veiksmīgi parsēts un validēts, notiek testu izveidošana pēc testu izveidošanas algoritma, kas tika aprakstīts 4.4.2. apakšnodaļā. Dotajā DSL izmantošanas piemērā XML faila satura apstrādes un testu izveidošanas rezultāts ir veiksmīgs. Līdz ar to tika uzģenerēti 4 (četri) varianti (sk. 4.10. attēlu) atbilstoši ievadītajam kodam, kas ir redzams 4.15. attēlā. Visi izveidotie testa varianti parādās kursa pirmajā tēmā un pēc noklusējuma tiks paslēpti no studentiem. Noklikšķinot uz testa variantu, var redzēt testa variantā iestatījumus un iekļautos jautājumus.

▼ Topic 1



The screenshot displays a Moodle interface for 'Topic 1'. It contains four identical quiz items, each represented by a pink square icon with a white checkmark. To the right of the icon, the text 'QUIZ' is in blue, and 'Eksamens 2023' is in a lighter blue. Below this, a yellow banner with the text 'Hidden from students' is visible. The items are arranged vertically in a list.

4.15. att. – Uzģenerētie Moodle testa varianti

Questions

Questions: 11 | This quiz is open

Maximum grade 10.00

Save

Repaginate

Select multiple items

Total of marks: 20.00

 ☐ Shuffle 

Page 1

Add 

 1

 **Cepeškrāsns mpī** Cepeškrāsns mpī

Always latest 

2.00 

Page 2

Add 

 2

 **Adreses lauki** Adreses lauki

Always latest 

2.00 

Page 3

Add 

 3

 **Radīt Studiju Programma** Radīt Studiju Programma

Always latest 

2.00 

Page 4

Add 

 4

 **Kursi Nav DatZ** Kursi Nav DatZ

Always latest 

2.00 

Page 5

Add 

 5

 **F(4)** F(4)

Always latest 

2.00 

Page 6

Add 

 6

 **a private b private in B** a private b private in B

Always latest 

2.00 

Page 7

Add 

 7

 **XSS** XSS

Always latest 

2.00 

Page 8

Add 

 8

 **Kļūda ieciklojas** Kļūda ieciklojas

Always latest 

2.00 

Page 9

Add 

 9

 **Xamarin** Xamarin

Always latest 

1.00 

Page 10

Add 

 10

 **Razor** Razor

Always latest 

1.00 

Page 11

Add 

 11

 **Vai palaists Process Notepad.exe** Vai palaists Proces...

Always latest 

2.00 

4.16. att. – Uzģenerētā testa 1. varianta uzdevumi

Kā jau tika minēts 4.4.2. apakšnodaļā, DSL interpretatora algoritms izstrādāts tā, lai uzģenerētie testa varianti pēc iespējas vairāk atšķirtos. 4.16. attēlā var redzēt jautājumus, kas tika iekļauti testa 1. variantā. Savukārt 2., 3., 4. testa varianta uzdevumus var redzēt 6., 7., 8. pielikumā.

Sakarā ar to, ka tika iestatīta piekļuve testam, ir nepieciešams to pārbaudīt testa iestatījumu sadaļā. 4.17. attēlā var redzēt, ka testa 1. variantam ir piekļuves ierobežojums – tests pieejams tikai lietotājam ar identifikatoru “jb1”. Pēc testu izveides manuāli jāpārbauda citi iestatījumi: piekļuves laiks testam, maksimālā atzīme, minimālā atzīme testa nokārtošanai utt.

▼ Restrict access

Access restrictions

The screenshot shows a web interface for setting access restrictions. At the top, it says "Access restrictions". Below this, there is a rule configuration area. The rule is set to "Student" (indicated by an eye icon) and "must match the following". Inside a box, there is a condition: "User profile field" is "ID number" and "is equal to" the value "jb1". There is a close button (X) and an "Add restriction..." button at the bottom of the configuration area.

4.17. att. – Testa 1. varianta piekļuves ierobežojumi

Pēc DSL izmantošanas piemēra var secināt, ka pašlaik DSL izmantošanas scenārijs nav ideāls. Bakalaura ietvaros tika īstenots prototips, kura pamatā ir XML faila manuālā augšupielāde. Tas prasa papildu darbības – patstāvīgi ģenerēt XML failu, uzģenerēto failu saglabāt un augšupielādēt formā. Otrs izstrādātās domēna specifiskās valodas trūkums ir tāds, ka valodas izmantošana ir iespējama tikai JetBrains MPS vidē. Tostarp valodu var uzlabot, pievienojot citus testa iestatījumus, kā arī paplašināt, izstrādājot jauno funkcionalitāti.

REZULTĀTI UN DISKUSIJA

Bakalaura darba ietvaros tika izstrādāta domēna specifiskā valoda Moodle testu izveidošanai. Domēna specifiskā valoda tika realizēta JetBrains MPS rīkā. Jaunā valoda ļauj:

- izveidot vairākus testus un to variantus;
- atlasīt jautājumus no Moodle kursa jautājumu bankas, izmantojot nosaukumu apakšvirknes un tagus;
- sadalīt atlasītos jautājumus variantos tā, lai jautājumi variantos atšķirtos;
- pievienot iestatījumus katram testa variantam.

Tika izstrādāts JetBrains MPS valodas definēšanas modulis, kurš sevī ietver domēna specifiskās valodas struktūras, redaktora un ģeneratora aspektu. Tika definēta domēna specifiskās valodas moduļu transformācija XML formāta moduļos. Uzģenerēto XML failu var izmantot turpmākai testu izveidošanai Moodle platformā.

Mācību pārvaldības sistēmai Moodle tika izstrādāts jauns spraudnis, kas satur XML faila augšupielādes formu. Moodle spraudnī tika realizēts DSL interpretators, kurš balstoties uz augšupielādēto XML dokumenta saturu, izveido Moodle testus.

Tā kā bakalaura darba ietvaros tika izstrādāts domēna specifiskās valodas prototips, jaunā domēna specifiskā valoda var tikt uzlabota un paplašināta. Kā arī testu izveidošana prasa papildu darbības (uzģenerētā XML faila ģenerēšana, augšupielāde Moodle formā). Nākotnē var nodrošināt domēna specifiskās valodas integrēšanu Moodle platformā vai to savstarpējo komunikāciju.

SECĪNĀJUMI

Bakalaura darba ietvaros tika izpētīta Moodle arhitektūra (sk. 1. nodaļu), domēna specifiskās valodas jēdziens, to veidi un izstrādes rīki (sk. 2. nodaļu), JetBrains MPS rīks domēna specifisko valodu izstrādei (sk. 3. nodaļu).

Bakalaura darba praktiskajā daļā tika izstrādāta domēna specifiskā valoda Moodle testu izveidošanai (sk. 4. nodaļu). Jaunā domēna specifiskā valoda nodrošina ērtu sintaksi testu izveidošanas aprakstīšanai un izveidošanai. Izmantojot jauno domēna specifisko valodu, ir iespējams:

- ģenerēt vairākus testus un to variantus;
- atlasīt dažādus jautājumus no Moodle kursa jautājumu bankas, izmantojot nosaukumu apakšvirknes un tagus;
- iekļaut atlasītus jautājumus variantos tā, lai jautājumi variantos atšķirtos;
- pievienot iestatījumus katram testa variantam.

Domēna specifiskā valoda tika realizēta JetBrains MPS rīkā. JetBrains MPS rīkā tika definēts valodas definēšanas modulis, kas ietver valodas struktūras, redaktora un ģeneratora aspektu. Valodas definēšanas moduļa izstrādes rezultātā ir iespējams izmantot izstrādāto DSL JetBrains MPS vidē un ģenerēt XML failu atbilstoši DSL kodam. Tāpat tika izstrādāts Moodle spraudnis (sk. 4.4. apakšnodaļu), kas satur formu uzģenerētā XML faila augšupielādei. Jaunajā spraudnī tika realizēts DSL interpretators, kas ģenerē testus un to variantus Moodle kursā, balstoties uz augšupielādēto XML dokumenta saturu.

Bakalaura darba ietvaros tika īstenots domēna specifiskās valodas prototips, kura pamatā ir XML faila manuālā augšupielāde. Pēc DSL izmantošanas piemēra (sk. 4.5. apakšnodaļu) var secināt, ka jaunā DSL izmantošanas scenārijs nav ideāls, jo prasa papildu darbības – patstāvīgi ģenerēt XML failu, uzģenerēto failu saglabāt un augšupielādēt formā. Tāpat izstrādāto DSL var izmantot tikai JetBrains MPS vidē. Tomēr domēna specifiskā valoda paātrina un uzlabo Moodle testu izveidošanas procesu. Līdz ar to var tikt uzlabota un paplašināta ar jaunām iespējām.

Darba gaitā tika identificētas domēna specifisko valodu izstrādes priekšrocības un trūkumi. Galvenā domēna specifiskās valodas priekšrocība – ir iespējams efektīvi atrisināt risināmo problēmu. Otrkārt, izmantojot domēna specifisko valodu, ir iespējams automatizēt domēna lietotāju darbības. Piemēram, izveidotā valoda ļauj lietotājiem ietaupīt laiku, ļaujot veidot vairākus testus un to variantus Moodle mācību pārvaldības sistēmā. Tomēr ārējai domēna specifiskai valodai ir nepieciešams izstrādāt redaktoru, kurā ir iespējams izmantot izstrādāto valodu, un ir jānodrošina eksporta/importa funkcionalitāte to integrēšanai citā sistēmā.

Var secināt, ka, lai izveidotu efektīvu DSL:

- labi jāsaprot domēns;
- labi jāizprot problēma un tās risinājums;
- nepieciešamas labas modelēšanas, projektēšanas un DSL izstrādes prasmes.

Līdz ar to lēmums izstrādāt DSL ir jāpieņem rūpīgi un pārdomāti, iepriekš izpētot risināmo problēmu un novērtējot izstrādes izmaksas.

PATEICĪBAS

Autore izsaka lielu pateicību doc. Elīnai Kalniņai par bakalaura darba vadīšanu un lielisko atbalstu visa bakalaura darba rakstīšanas laikā. It īpaši par palīdzību interesantas tēmas izvēlē, sistemātiskām sapulcēm un vērtīgiem ieteikumiem.

IZMANTOTĀ LITERATŪRA UN AVOTI

- [1] «Moodle» Moodle.org, [Tiešsaiste]. Pieejams: <https://moodle.org/>. [Piekļūts 15 05 2023].
- [2] «Blackboard learn» Blackboard Inc, [Tiešsaiste]. Pieejams: <https://www.blackboard.com/en-mea/teaching-learning/learning-management/blackboard-learn>. [Piekļūts 28 05 2023].
- [3] «Canvas» Canvas Network, [Tiešsaiste]. Pieejams: <https://www.canvas.net/>. [Piekļūts 28 05 2023].
- [4] J. Babčenoka, «Specializēto valodu izmantošana tiešsaistes mācību sistēmās. Kursa darbs, Latvijas Universitātes datorzinātņu bakalaura studiju programmā. » Rīga, 2023.
- [5] «Moodle GitHub» Moodle, [Tiešsaiste]. Pieejams: <https://github.com/moodle/moodle>. [Piekļūts 14 04 2023].
- [6] «Moodle architecture» Moodle.org, [Tiešsaiste]. Pieejams: https://docs.moodle.org/dev/Moodle_architecture. [Piekļūts 08 04 2023].
- [7] «PHP» The PHP Group, [Tiešsaiste]. Pieejams: <https://www.php.net/>. [Piekļūts 29 05 2023].
- [8] «JavaScript» JavaScript, [Tiešsaiste]. Pieejams: <https://www.javascript.com/>. [Piekļūts 29 05 2023].
- [9] «Communication Between Components» Moodle.org, [Tiešsaiste]. Pieejams: https://docs.moodle.org/dev/Communication_Between_Components. [Piekļūts 08 04 2023].
- [10] «Moodle spraudņu katalogs» Moodle.org, [Tiešsaiste]. Pieejams: <https://moodle.org/plugins/index.php>. [Piekļūts 08 04 2023].
- [11] «Database transfer» Moodle.org, 01 07 2022. [Tiešsaiste]. Pieejams: https://docs.moodle.org/402/en/Database_transfer. [Piekļūts 06 05 2023].
- [12] «Moodle 4.0. database schema» Moodle, [Tiešsaiste]. Pieejams: <https://www.examulator.com/er/4.0/>. [Piekļūts 08 04 2023].
- [13] «Quiz_database_structure» Moodle.org, [Tiešsaiste]. Pieejams: https://docs.moodle.org/dev/Quiz_database_structure. [Piekļūts 07 04 2023].
- [14] «Moodle quiz_slots» Moodle, [Tiešsaiste]. Pieejams: https://www.examulator.com/er/4.0/tables/quiz_slots.html. [Piekļūts 15 04 2023].

- [15] «Data manipulation API» Moodle.org, 28 11 2022. [Tiešsaiste]. Pieejams: <https://moodledev.io/docs/apis/core/dml>. [Piekļūts 07 04 2023].
- [16] L. Bettini, «Domain-Specific Languages with Xtext and Xtend» *Implementing Domain-Specific Languages with Xtext and Xtend*, 2013.
- [17] D. H. Lorenz un B. Rosenan, «Cedalion - A Language Oriented Programming Language (Extended Abstract)*» *Conference on Object-Oriented Programming Systems, and Applications*, 2011.
- [18] M. Fowler, «Chapter 2: Using Domain-Specific Languages» *Domain Specific Languages*, Addison-Wesley Professional, 2010, p. 640.
- [19] «Eclipse» Eclipse Foundation, [Tiešsaiste]. Pieejams: <https://www.eclipse.org/>. [Piekļūts 21 03 2023].
- [20] «Eclipse Modeling Framework (EMF)» Eclipse Foundation, [Tiešsaiste]. Pieejams: <https://www.eclipse.org/modeling/emf/>. [Piekļūts 01 05 2023].
- [21] «What Can You Do with Sirius?» Eclipse Foundation, [Tiešsaiste]. Pieejams: <https://www.eclipse.org/sirius/gallery.html>. [Piekļūts 23 03 2023].
- [22] D. Julien un M. Frédéric, «Mindstorms Robot Tutorial» Eclipse Foundation, 24 06 2021. [Tiešsaiste]. Pieejams: <https://wiki.eclipse.org/Sirius/Tutorials/Mindstorms>. [Piekļūts 28 05 2023].
- [23] J. Yue, «Transition from EBNF to Xtext» PSRC@MoDELS, Nanjing, China, 2014.
- [24] M. V. Sven Efftinge, «oAW xText: A framework for textual DSLs» Sven Efftinge, Markus Völter, 2006.
- [25] J.-P. Tolvanen, R. Pohjonen un S. Kelly, «Advanced Tooling for Domain-Specific Modeling: MetaEdit+» 2007.
- [26] «MetaCase» MetaCase, 2023. [Tiešsaiste]. Pieejams: <https://www.metacase.com/>. [Piekļūts 01 05 2023].
- [27] «JetBrains MPS» JetBrains s.r.o., [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/mps/>. [Piekļūts 15 03 2023].
- [28] «JetBrains MPS Abstract Syntax Tree (AST)» JetBrains s.r.o., 31 08 2021. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/basic-notions.html>. [Piekļūts 2023 03 16].
- [29] A. Balderas, M. Palomo-Duarte, J. M. Dodero un R.-R. Iván, «A Domain Specific Language for Online Learning Competence Assessments» *International Journal of Engineering Education*, sēj. 31, pp. 851-862, 2015.

- [30] C. E. MONTENEGRO-MARÍN, J. M. CUEVA-LOVELLE, O. SANJUÁN-MARTÍNEZ un V. GARCÍA-DÍAZ, «DOMAIN SPECIFIC LANGUAGE FOR THE GENERATION OF LEARNING MANAGEMENT SYSTEMS MODULES. » *Journal of Web Engineering*, sēj. 11, pp. 023-050, 2012.
- [31] S. Meacham, V. Pech un D. Nauck, «AdaptiveVLE: An Integrated Framework for Personalized Online Education Using MPS JetBrains Domain-Specific Modeling Environment» *IEEE Access*, sēj. 8, pp. 184621-184632, 2020.
- [32] A. Cicchetti, F. Ciccozzi, A. Pierantonio un A. Bucchiarone, «MPS domēni» *Domain-Specific Languages in Practice: with JetBrains MPS*, 2021, p. 3.
- [33] E. Tomáš, R. Firment, J. Saksa, M. Wirth un D. Zeman, «C# Base Language for MPS» 2019. [Tiešsaiste]. Pieejams: https://www.ksi.mff.cuni.cz/sw-projekty/zadani/cs4mps_spec.pdf. [Piekļūts 28 04 2023].
- [34] «MPS project structure» JetBrains MPS, 22 09 2021. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/mps-project-structure.html>. [Piekļūts 28 05 2023].
- [35] «JetBrains MPS Structure» JetBrains s.r.o., 31 05 2022. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/structure.html>. [Piekļūts 29 05 2023].
- [36] «JetBrains MPS Editor» JetBrains s.r.o., 03 10 2021. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/editor.html>. [Piekļūts 28 05 2023].
- [37] «JetBrains MPS Constraints» JetBrains s.r.o., 16 12 2023. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/constraints.html#defaultconcreteconcept>. [Piekļūts 28 05 2023].
- [38] «JetBrains MPS Behavior» JetBrains s.r.o., 23 03 2021. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/behavior.html>. [Piekļūts 28 05 2023].
- [39] «JetBrains MPS Generator» JetBrains s.r.o., 19 09 2022. [Tiešsaiste]. Pieejams: <https://www.jetbrains.com/help/mps/mps-generator.html#overview>. [Piekļūts 29 05 2023].

PIELIKUMI

1. pielikums. DSL koncepti

Koncepts	Definējums
QuizesBlock	<pre> concept QuizesBlock extends BaseConcept implements <none> instance can be root: true alias: <no alias> short description: <no short description> properties: << ... >> children: Quizes : Quiz[1..n] </pre>
Quiz	<pre> concept Quiz extends BaseConcept implements <none> instance can be root: false alias: <no alias> short description: <no short description> properties: title : string children: ExercisesBlock : ExercisesBlock[1] VariantsBlock : VariantBlock[1..n] SettingsBlock : SettingsBlock[1] references: << ... >> </pre>

Koncepts	Definējums
ExercisesBlock	<pre> concept ExercisesBlock extends BaseConcept implements <none> instance can be root: false alias: <no alias> short description: <no short description> properties: questionCategory : string children: Exercises : Exercise[0..n] </pre>
Exercise	<pre> abstract concept Exercise extends BaseConcept implements <none> instance can be root: false alias: <no alias> short description: <no short description> properties: count : ExerciseCountType </pre>
Exercise WithSubname	<pre> concept ExerciseWithSubname extends Exercise implements <none> instance can be root: false alias: contains subname short description: <no short description> properties: subname : string </pre>

Koncepts	Definējums
Exercise WithTag	<pre> concept ExerciseWithTag extends Exercise implements <none> instance can be root: false alias: contains tag short description: <no short description> properties: tag : string </pre>
VariantBlock	<pre> concept VariantBlock extends BaseConcept implements <none> instance can be root: false alias: <no alias> short description: <no short description> properties: usersSeparator : QuizUserSeparatorEnum children: Users : User[0..n] references: << ... >> </pre>
User	<pre> concept User extends BaseConcept implements INamedConcept instance can be root: false alias: <no alias> short description: <no short description> properties: field : ProfileFieldEnum condition : FieldConditionEnum children: << ... >> references: << ... >> </pre>

Koncepts	Definējums
SettingsBlock	<pre> concept SettingsBlock extends BaseConcept implements <none> instance can be root: false alias: <no alias> short description: <no short description> properties: shuffle : boolean attempts : AttemptsType children: << ... >> references: << ... >> </pre>

2. pielikums. DSL konceptu redaktori

Koncepts	Redaktors
Quizes Block	<pre> <default> editor for concept QuizesBlock node cell layout: [/ Create new quizzes (/ % Quizes % /) /empty cell: <default> /]</pre>
Quiz	<pre> <default> editor for concept Quiz node cell layout: [/ empty create a quiz [> set title " { title } " <] % ExercisesBlock % (/ % VariantsBlock % /) /empty cell: <default> % SettingsBlock % /]</pre> <p>inspected cell layout:</p> <pre> <choose cell model></pre>
Exercises Block	<pre> <default> editor for concept ExercisesBlock node cell layout: [/ include random exercises [> from question category " { questionCategory } " <] (/ % Exercises % /) /empty cell: [> <constant> <constant> do not include exercises <] /]</pre> <p>inspected cell layout:</p> <pre> <choose cell model></pre>
Exercise With Subname	<pre> <default> editor for concept ExerciseWithSubname node cell layout: [> <constant> <constant> { count } with subname " { subname } " <]</pre> <p>inspected cell layout:</p> <pre> <choose cell model></pre>

Koncepts	Redaktors
Exercise WithTag	<pre> <default> editor for concept ExerciseWithTag node cell layout: [> <constant> <constant> { count } with tag " { tag } " <] inspected cell layout: <choose cell model> </pre>
Variant Block	<pre> <default> editor for concept VariantBlock node cell layout: [/ [> create a variant for users ({ usersSeparator }) <] [/ (/ % Users % /) /empty cell: [> <constant> <constant> all course users <] /] /] </pre> <p>inspected cell layout:</p> <p><choose cell model></p>
User	<pre> <default> editor for concept User node cell layout: [> <constant> <constant> which { field } { condition } " { name } " <] inspected cell layout: <choose cell model> </pre>
Settings Block	<pre> <default> editor for concept SettingsBlock node cell layout: [/ set settings [> <constant> <constant> { attempts } attempts <] [> <constant> <constant> (if true: shuffle questions if) <constant> \$swing component\$ <] false: do not shuffle questions <constant> /] </pre> <p>inspected cell layout:</p> <p><choose cell model></p>

3. pielikums. DSL pārveidošanas veidnes

Koncepts	Veidne
QuizesBlock	<pre> [root template input QuizesBlock] xml map_QuizesBlock.xml <no prolog> <moodleTests> \$COPY_SRCL\$ [] </moodleTests> </pre>
Quiz	<pre> template reduce_Quiz input Quiz parameters << ... >> content node: <TF [<moodleTest> <title>\$[]</title> \$COPY_SRC\$ [] \$COPY_SRC\$ [] \$COPY_SRC\$ [] </moodleTest>] TF> </pre>
Exercises Block	<pre> template reduce_ExercisesBlock input ExercisesBlock parameters << ... >> content node: <TF [<exercises> <questionCategory>\$[]</questionCategory> \$COPY_SRCL\$ [] </exercises>] TF> </pre>

Koncepts	Veidne
Exercise WithSubname	<pre> template reduce_ExerciseWithSubname input ExerciseWithSubname parameters << ... >> content node: <TF [<withSubname>] TF> <count>\${ }</count> <subname>\${ }</subname> </withSubname> </pre>
Exercise WithTag	<pre> template reduce_ExerciseWithTag input ExerciseWithTag parameters << ... >> content node: <TF [<withTag>] TF> <count>\${ }</count> <tag>\${ }</tag> </withTag> </pre>
VariantBlock	<pre> template reduce_VariantBlock input VariantBlock parameters << ... >> content node: <TF [<quizVariant usersSeparator="\${ }">] TF> \$COPY_SRCL\$[] </quizVariant> </pre>

Koncepts	Veidne
User	<pre> template reduce_User input User parameters << ... >> content node: <TF> [<quizUser> TF> <searchField>\${}</searchField> <condition>\${}</condition> <name>\${}</name> </quizUser> </pre>
SettingsBlock	<pre> template reduce_SettingsBlock input SettingsBlock parameters << ... >> content node: <TF> [<settings> TF> <shuffle>\${}</shuffle> <attempts>\${}</attempts> </settings> </pre>

4. pielikums. Kursa jautājumu bankas piemērs

Nr.	Jautājums	Tags	Punktu skaits
1.	a private b private in B	PrivatePublic	2
2.	a private b private in C	PrivatePublic	2
3.	a private b public in C	PrivatePublic	2
4.	a private b public in C slēpti	PrivatePublic	2
5.	a public b private in C	PrivatePublic	2
6.	Bezvadu Austiņas mpī	MPī	2
7.	Cepeškrāsns mpī	MPī	2
8.	Galda Lampa mpī	MPī	2
9.	Gludeklis mpī	MPī	2
10.	Dependency Property	Teorija	2
11.	XSS	Teorija	2
12.	NET 5	Teorija	2
13.	$f(1,5)$	Funkcija	2
14.	$F(4)$	Funkcija	2
15.	$F(8)$	Funkcija	2
16.	$G(4)$	Funkcija	2
17.	$G(8)$	Funkcija	2
18.	Kursi	LINQ	2
19.	Kursi 6cp	LINQ	2
20.	Kursi Matemātika	LINQ	2
21.	Kursi Nav DatZ	LINQ	2
22.	Kursu kodi, 2 CP	LINQ	2
23.	Ledusskapis mpī	MPī	2
24.	AdreSES lauki	Kods	2
25.	PilnaAdrese	Kods	2
26.	Radīt Studiju Programma	Kods	2
27.	Students uztaisīt, 2 konstruktori	Kods	2
28.	Students, 3 īpašības	Kods	2
29.	StudijuProgramma	Kods	2
30.	Uztaisīt studentu, īpašības	Kods	2

Nr.	Jautājums	Tags	Punktu skaits
31.	Interfeiss Persona	Kods	2
32.	Klase Students ar interfeisu	Kods	2
33.	Darba virsmas	Tests	1
34.	Klasei priekšā	Tests	1
35.	LINQ	Tests	1
36.	Metodei priekšā	Tests	1
37.	Razor	Tests	1
38.	Tīmekļa	Tests	1
39.	Title WPF	Tests	1
40.	Unity	Tests	1
41.	Xamarin	Tests	1
42.	XAML	Tests	1
43.	aa.dll Course	Advanced	2
44.	aa.dll Students	Advanced	2
45.	Asinhroni findPrime	Advanced	2
46.	Asinhroni GCD	Advanced	2
47.	Kļūda Ieciklojas	Advanced	2
48.	Vai palaists Process Notepad.exe	Advanced	2

5. pielikums. Uzģenerētā XML faila saturs

```
<moodleTests>
  <moodleTest>
    <title>Eksamens 2023</title>
    <exercises>
      <questionCategory>Eksamens</questionCategory>
      <withTag>
        <count>1</count>
        <tag>MPĪ</tag>
      </withTag>
      <withTag>
        <count>2</count>
        <tag>Kods</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>LINQ</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>Funkcija</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>PrivatePublic</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>Teorija</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>Advanced</tag>
      </withTag>
      <withTag>
        <count>2</count>
        <tag>Tests</tag>
      </withTag>
      <withTag>
        <count>1</count>
        <tag>Advanced</tag>
      </withTag>
    </exercises>
    <quizVariant usersSeparator="or">
      <user>
        <searchField>idnumber</searchField>
        <condition>isequalTo</condition>
        <name>jb1</name>
      </user>
    </quizVariant>
    <quizVariant usersSeparator="or">
      <user>
        <searchField>idnumber</searchField>
        <condition>isequalTo</condition>
        <name>jb2</name>
      </user>
    </quizVariant>
  </moodleTest>
</moodleTests>
```

```

    </user>
</quizVariant>
<quizVariant usersSeparator="or">
  <user>
    <searchField>idnumber</searchField>
    <condition>isequalto</condition>
    <name>jb3</name>
  </user>
</quizVariant>
<quizVariant usersSeparator="or">
  <user>
    <searchField>idnumber</searchField>
    <condition>isequalto</condition>
    <name>jb4</name>
  </user>
</quizVariant>
<settings>
  <shuffle>>false</shuffle>
  <attempts>1</attempts>
</settings>
</moodleTest>
</moodleTests>

```

6. pielikums. Testa 2. varianta uzdevumi

Questions

Questions: 11 | This quiz is open


Maximum grade 10.00

Save

Repaginate

Select multiple items

Total of marks: 20.00

 Shuffle ?

Page 1

+

1

••

⚙

Bezvadu Austiņas mpī

Bezvadu Austiņas mpī

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 2

+

2

••

⚙

Interfeiss Persona

Interfeiss Persona

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 3

+

3

••

⚙

PilnaAdrese

PilnaAdrese

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 4

+

4

••

⚙

Kursi 6cp

Kursi 6cp

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 5

+

5

••

⚙

f(1,5)

f(1,5)

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 6

+

6

••

⚙

a private b private in C

a private b private in C

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 7

+

7

••

⚙

NET 5

NET 5

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 8

+

8

••

⚙

Vai palaists Process Notepad.exe

Vai palaists Proces...

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

Page 9

+

9

••

⚙

Darba virsmas

Darba virsmas

Always latest

🔍

🗑

1.00

⌵

⌶

Add

▼

Page 10

+

10

••

⚙

Unity

Unity

Always latest

🔍

🗑

1.00

⌵

⌶

Add

▼

Page 11

+

11

••

⚙

aa.dll Course

aa.dll Course

Always latest

🔍

🗑

2.00

⌵

⌶

Add

▼

61

7. pielikums. Testa 3. varianta uzdevumi

Questions

Questions: 11 | This quiz is open


Maximum grade 10.00

Save

Repaginate

Select multiple items

Total of marks: 20.00

 Shuffle ?


Page 1

Add ▾

+

1


••





Ledusskapis mpī

Ledusskapis mpī

Always latest ▾





2.00 


Page 2

Add ▾

+

2


••





Students uztaisīt, 2 konstruktori

Students uztaisīt, 2 ...

Always latest ▾





2.00 


Page 3

Add ▾

+

3


••





Students, 3 īpašības

Students, 3 īpašības

Always latest ▾





2.00 


Page 4

Add ▾

+

4


••





Kursi Matemātika

Kursi Matemātika

Always latest ▾





2.00 


Page 5

Add ▾

+

5


••





G(8)

G(8)

Always latest ▾





2.00 


Page 6

Add ▾

+

6


••





a public b private in C

a public b private in C

Always latest ▾





2.00 


Page 7

Add ▾

+

7


••





Dependency Property

Dependency Property

Always latest ▾





2.00 


Page 8

Add ▾

+

8


••





Asinhroni GCD

Asinhroni GCD

Always latest ▾





2.00 


Page 9

Add ▾

+

9


••





Klasei priekšā

Klasei priekšā

Always latest ▾





1.00 


Page 10

Add ▾

+

10


••





XAML

XAML

Always latest ▾





1.00 


Page 11

Add ▾

+

11


••





aa.dll Students

aa.dll Students

Always latest ▾





2.00 

8. pielikums. Testa 4. varianta uzdevumi

Questions

Questions: 11 | This quiz is open


Maximum grade 10.00

Save

Repaginate

Select multiple items

Total of marks: 20.00

 Shuffle ?

Page 1

+

1

⚙

Gludeklis mpī Gludeklis mpī

Always latest

2.00

Page 2

+

2

⚙

Uztaisīt studentu, īpašības Uztaisīt studentu, īpašības

Always latest

2.00

Page 3

+

3

⚙

StudijuProgramma StudijuProgramma

Always latest

2.00

Page 4

+

4

⚙

Kursi Kursi

Always latest

2.00

Page 5

+

5

⚙

G(4) G(4)

Always latest

2.00

Page 6

+

6

⚙

a private b public in C a private b public in C

Always latest

2.00

Page 7

+

7

⚙

XSS XSS

Always latest

2.00

Page 8

+

8

⚙

aa.dll Course aa.dll Course

Always latest

2.00

Page 9

+

9

⚙

Title WPF Title WPF

Always latest

1.00

Page 10

+

10

⚙

Tīmekļa Tīmekļa

Always latest

1.00

Page 11

+

11

⚙

Asinhroni GCD Asinhroni GCD

Always latest

2.00

63

9. pielikums. Izstrādātā XML faila augšupielādes forma

LMSMoodle

HomeDashboardMy coursesSite administration

?

Edit mode

.NET course

CourseSettingsParticipantsGradesReportsMore

XML uploading form

Upload xml with quizzes definitions

Choose a file...

moodleTests.xml

Accepted file types:
application/xml .xml

Create

Required

10. pielikums. Jautājumu atlasīšana, izmantojot nosaukuma apakšvirkni

```
function get_questions_from_categories_and_subname($categoryids, $subname) {
    global $DB;

    list($qcsql, $qcparams) = $DB->get_in_or_equal($categoryids, SQL_PARAMS_NAMED, 'qc');
    $qcparams['readystatus'] = \core_question\local\bank\question_version_status::QUESTION_STATUS_READY;
    $qcparams['subname'] = '%'.$subname.'%';

    $sql = "SELECT q.id
            FROM (
                SELECT q.name, MAX(qv.version) maxqversion
                FROM {question} q
                JOIN {question_versions} qv ON qv.questionid = q.id
                JOIN {question_bank_entries} qbe ON qbe.id = qv.questionbankentryid
                WHERE qbe.questioncategoryid {$qcsql}
                AND q.parent = 0
                AND qv.status = :readystatus
                AND q.name LIKE :subname
                GROUP BY q.name
            ) subq
            JOIN mdl_question q ON q.name = subq.name
            JOIN mdl_question_versions qv ON qv.questionid = q.id AND qv.version = subq.maxqversion
            ORDER BY q.name";
    $retrieved_questions = $DB->get_records_sql_menu($sql, $qcparams);

    return array_keys($retrieved_questions);
}
```

11. pielikums. Jautājumu atlasīšana, izmantojot tagu

```
function get_questions_from_categories_and_tag($categoryids, $tag) {
    global $DB;

    list($qcsql, $qcparams) = $DB->get_in_or_equal($categoryids, SQL_PARAMS_NAMED, 'qc');
    $qcparams['readystatus'] = \core_question\local\bank\question_version_status::QUESTION_STATUS_READY;
    $qcparams['questionitemtype'] = 'question';
    $qcparams['questioncomponent'] = 'core_question';
    $qcparams['tag'] = $tag;

    $sql = "SELECT q.id
    FROM (
        SELECT q.name, MAX(qv.version) maxqversion
        FROM {question} q
        JOIN {question_versions} qv ON qv.questionid = q.id
        JOIN {question_bank_entries} qbe ON qbe.id = qv.questionbankentryid
        WHERE qbe.questioncategoryid {$qcsql}
        AND q.parent = 0
        AND qv.status = :readystatus
        AND q.id IN (SELECT ti.itemid
                     FROM {tag_instance} ti
                     JOIN {tag} t ON t.id = ti.tagid
                     WHERE ti.itemtype = :questionitemtype
                          AND ti.component = :questioncomponent
                          AND t.rawname = :tag)
        GROUP BY q.name
    ) subq
    JOIN mdl_question q ON q.name = subq.name
    JOIN mdl_question_versions qv ON qv.questionid = q.id AND qv.version = subq.maxqversion
    ORDER BY q.name";

    $retrieved_questions = $DB->get_records_sql_menu($sql, $qcparams);

    return array_keys($retrieved_questions);
}
```