Федеральное государственное автономное
образовательное учреждение
высшего образования
"СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ"

Институт космических и информационных технологий
институт

Кафедра вычислительной техники
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ВТ
_____А.И.Легалов
"___"_____20____г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Подсистема формирования графических элементов для системы активного тестирования
тема

09.04.01 Кафедра вычислительной техники
код и наименование направления

09.04.01.02 Информационное и программное обеспечение САПР
код и наименование магистерской программы

Руководитель	_____	проф., д-р. техн. наук	С.А.Бронов
	подпись, дата	*должность, ученая степень*	*инициалы, фамилия*
Выпускник	_____		Д.М.Севостьянов
	подпись, дата		*инициалы, фамилия*
Рецензент	_____	доц., канд. техн. наук	Ю.В.Краснобаев
	подпись, дата	*должность, ученая степень*	*инициалы, фамилия*
Нормоконтролер	_____	доц., канд. техн. наук	В.И.Иванов
	подпись, дата	*должность, ученая степень*	*инициалы, фамилия*

Красноярск 2017

АННОТАЦИЯ

Магистерская диссертация " Система формирования графических элементов для системы активного тестирования" выполнена в НУЛ САПР СФУ.

Объект и предмет исследования — принципы формирования графических элементов для системы активного тестирования.

Цель работы — создание подсистемы для системы автоматического тестирования студентов, направленной на создание графических элементов, изменение их характеристик, а также сохранение готового элемента на сервис, для дальнейшего его использования в редакторе тестов.

На языке программирования Java с использованием библиотек Swing и AWT разработано приложение для создания графических элементов из списка графических примитивов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность. В настоящее время образование тесно связано с всемирной сетью. Несмотря на то, что обучение традиционными методами остаётся наилучшим способом донести до обучающегося предмет, создание собственного электронного ресурса обучения является довольно актуальной задачей. Это не только хороший тон для современных учебных заведений, но и способ упростить учебный процесс, в плане документооборота, донесения информации, и самоорганизации обучающегося. Однако создание такого ресурса остаётся, в большинстве случаев, экспериментальной мерой, направленной на изучение способов автоматизации учебного процесса. Также отсутствует единая форма электронного образовательного курса для разных учебных заведений. Высшие учебные заведения зачастую пытаются предложить свою форму системы электронных изданий, т.е. форму реализации, различные способы проверки заданий, способа их выполнения, а зачастую ограничивается обменом документов между обучающимся и преподавателем, с последующей оценкой знаний. Для начала, стоит дать определение электронного образовательного курса.

Электронный образовательный курс (ЭОК) — это образовательное электронное издание или ресурс для поддержки учебного процесса в учреждениях общего, специального, профессионального образования, а также для самообразования в рамках учебных программ, в том числе нацеленных на непрерывное образование[1]. ЭОК позволяет выполнять все основные методические функции электронных изданий:

– справочно-информационные;

– контролирующие;

– функции тренажера;

– имитационные;

– моделирующие;

– демонстрационные.

Исходя из определения, ЭОК — это довольно сложная система, способная поглотить традиционные методы обучения, и в идеальном случае, осуществлять контроль знаний и самостоятельное обучение на дому. Но как было сказано выше, зачастую не все функции таких курсов реализованы, и учебные заведения осуществляют обучение в «гибридном режиме», то есть подача материала осуществляется напрямую от преподавателя, контрольные работы проводятся в «бумажном» режиме, и лишь традиционные и адаптивные тесты проводятся в электронном варианте.

Создание подобных систем тестирования задача, которую сложно переоценить. Движение в сторону создания электронного тестирования поможет значительно увеличить производительность учебного процесса, а также улучшить его качества[9].

В свою очередь системы активного тестирования в качестве средств проверки знаний являются очень редким явлением в сфере оценки знания учащихся. Подобные системы способны проверить правильность ответа для заданий, построенных методом объединения графических образов. При этом порядок объединения будет основан на теоретическом правиле, которое в свою очередь может быть выполнено различными способами. Возможности подобных систем превосходят возможности обычных электронных тестов[10].

Система активного тестирования, разрабатываемая коллективом НУЛ САПР, предполагает наличие трёх подсистем. Данная работа посвящена созданию подсистемы, направленной на формирование графических элементов для системы активного тестирования. В рамках разработки сформировано несколько задач, необходимых для функционирования подсистемы. А именно:

– создание графического интерфейса, способного предоставить базовые функции для работы с графическими примитивами;

– представить список графических примитивов и их параметров в виде XML файла. Данный список в дальнейшем будет представлять собой единый элемент, использующийся для редактора тестов;

– обеспечить сохранение характеристик вновь созданного элемента, необходимых для использования их в редакторе;

– обеспечить возможность сохранения и загрузки данного списка в графический интерфейс для редактирования, или использования в качестве основы;

– создать скрипт, необходимый для загрузки элементов в общую базу данных.

Объектом исследования является способ представления графических элементов в виде, необходимом для подсистемы формирования графических образов, являющихся частями системы активного тестирования.

Предметом исследования формат представления графического элемента, необходимого для создания активного теста, в виде набора графических примитивов, объединённых в единый файл.

Объектом разработки является приложение, позволяющее создавать графические элементы в необходимом формате для дальнейшего использования в подсистеме формирования графических образов.

Цель работы — разработка и реализация системы формирования графических элементов для системы активного тестирования.

Задачи работы:

1 Определение функционала реализуемого приложения и требования формату предоставляемых данных.

2 Выбор методов и инструментальных средств, для разработки подсистемы формирования графических элементов.

3 Разработка способа конвертации графических данных в формат, необходимый для системы формирования графических образов.

4 Создание рабочей среды, необходимой для манипуляции пользователем графическими примитивами, и создания графического элемента.

Основная идея работы разработка программы-редактора для системы активного тестирования студентов, направленной на создание графических элементов. Функционал системы должен предполагать визуализацию графических примитивов, изменение их характеристик (добавление, удаление, измене-

ние параметров), а также предоставлять возможность сохранения готового элемента в формате, удобного для дальнейшего его использования в подсистеме формирования графических образов.

Методы, инструментальные средства, технологии разработки.

В качестве языка программирования был выбран Java, как популярный и гибкий инструмент для создания приложений. В качестве набора библиотек используется Swing, AWT, а также библиотеки для парсинга XML файлов[14].

В разделе 1 рассмотрена проблематика тестирования в целом, виды тестирования, его достоинства и недостатки. Проведён обзор некоторых систем тестирования, по результату сделан вывод. Были приведены требования к приложению, а также представлено описание средств разработки.

В разделе 2 описаны составляющие разрабатываемой системы активного тестирования. Также был описан формат конечного результата работы программы, классы, использующиеся для создания элемента.

В приложении А описан листинг записи и отрисовки примитивов в массив.

В приложении Б описан листинг парсинга графических примитивов и записи в XML документ.

В приложении В описаны основные принципы работы приложения, такие как создание элемента, перемещение, поворот, и т.д.

В приложении Г описаны правила описания графических элементов (XML теги, формат XML документа).

1 Проблематика тестирования и обзор существующих систем.

1.1 Тестирование. Достоинства и недостатки.

Проблематика электронных курсов заключается в способах передачи информации между студентом и преподавателем, а конкретно, способах формирования ответа на задание. Существует не так много способов проверки ответов на задания, используемых в подобных интернет ресурсах. Зачастую, в качестве проверки усвоенных студентом знаний используют тест.

Традиционное тестирование является выбором студентом одного или нескольких вариантов ответа на поставленный вопрос, и в зависимости от его выбора, можно определить, усвоил ли обучающийся учебный материал. В качестве плюсов теста, как средства контроля знаний, можно выделить:

– элементарность задания. В целом, если тест, включает в себя вопрос и несколько вариантов ответов, совершить ошибку в нём довольно сложно, и человек, не ответивший на него, скорее всего, не знает правильного ответа;

– упрощённый контроль усвоения материала**.** Результаты теста могут быть использованы в качестве исследования статистики по какому-либо вопросу. Например, статистику усвоения материала, статистику посещения теста, а также, корректность самого теста. Если подавляющее большинство из тех, кто прошёл тест, получил неудовлетворительную оценку за отдельно взятый вопрос, это может означать, что задание составлено некорректно (Это верно для сложных электронных тестов, состоящих из сотни вопросов, или вопросов, для которых случайно выбираются ответы, к примеру, 3 из возможных 10 вариантов);

– сокращённое время на проверку знаний. Для электронных вариантов тестов, время на проверку сокращается до минимума, а также, можно в доступной форме для экзаменатора вывести на экран ошибки обучающегося, чтобы обсудить их лично, или впоследствии, сделать упор на изучение материала, в котором обучающиеся допустили наибольшее число ошибок;

– автоматический контроль над ходом теста. В электронный тест, возможно, встроить методы контроля над студентами, такие как лимит времени, а также, возможность помощи обучающемуся, например, подсказки.

Несмотря на указанные плюсы, данный вид проверки знаний имеет множество недостатков, связанных в основном, с организационной деятельностью, такие как:

– сложность создания. Создание тестов — процесс монотонный и затратный по времени. Электронные тесты могут облегчить ситуацию, предложенным вариантом ответа (к примеру, расположенные в случайном порядке варианты ответа);

– вероятность ошибки. В тестах, состоящих из тысяч вопросов, могут находиться десятки ошибок, которые экзаменатор не может проконтролировать, и это может привести к получению более низкой оценки знаний студента;

– угадывание ответов. В небольших традиционных тестах есть вероятность того, что обучающийся может угадать вариант ответа;

– незащищённость. Одним из серьёзных недостатков электронных тестов — возможность получить базу данных с ответами на тест, а также сбор информации о вариантах ответов от студентов, уже прошедших его. Если ответы на тесты оказываются в свободном доступе — смысл тестирования сводится к нулю (верно для тестов с буквенными и цифровыми вариантами ответа).

Несмотря на то, что тест — основной вид тестирования пройденного материала в электронных курсах, для некоторых дисциплин имеется необходимость в ином виде проверки знаний. К примеру, ответом на задачу может служить какая-либо геометрическая фигура, или собранная электрическая цепь. Данный вид ответа может быть использован в тестировании (к примеру, вариантами ответа могут быть изображения схем), но например, электрическая цепь, служащая для одной и той же цели, может быть выполнена несколькими вариантами построения. То есть, с увеличением количества элементов, количество вариантов ответа увеличивается многократно. Данную проблему можно решить обычным составлением варианта ответа на бумаге и консультацией с

преподавателем, но это, в свою очередь, увеличивает затраты времени преподавателя, увеличивает количество документации.

Система тестирования — это инструмент, широко применяющийся не только в различных учебных заведениях, но и на промышленных объектах для контроля знаний персонала. Также, системы тестирования используются для проведения экзаменов в госучреждениях, для вынесения решения о допуске или не допуске персонала к выполнению обязанностей и т.д. Современные системы электронного тестирования являются готовыми решениями для внедрения в различные организации. Ниже представлены примеры таких систем тестирования.

1.2 Обзор существующих систем электронного тестирования

1.2.1 Система тестирования INDIGO

INDIGO — это профессиональный инструмент автоматизации процесса тестирования и обработки результатов, который предназначен для решения широкого спектра задач[5]:

– тестирование и контроль знаний учащихся;

– определение профессионального уровня сотрудников;

– проведение психологического тестирования;

– проведение опросов;

– организация олимпиад и конкурсов.

Система обладает множеством функциональных возможностей, из них можно выделить несколько основных:

– возможность работы системы как онлайн, так и офлайн;

– удобством работы администратора, управляющим процессом тестирования;

– поддержка мобильных устройств.

Система тестирования состоит из 2-х частей:

– windows-приложение – интерфейс администратора тестовой оболочки;

– Web-интерфейс – интерфейс пользователя тестовой оболочки.

Конструктор тестов позволяет задавать произвольное количество шкал результатов.

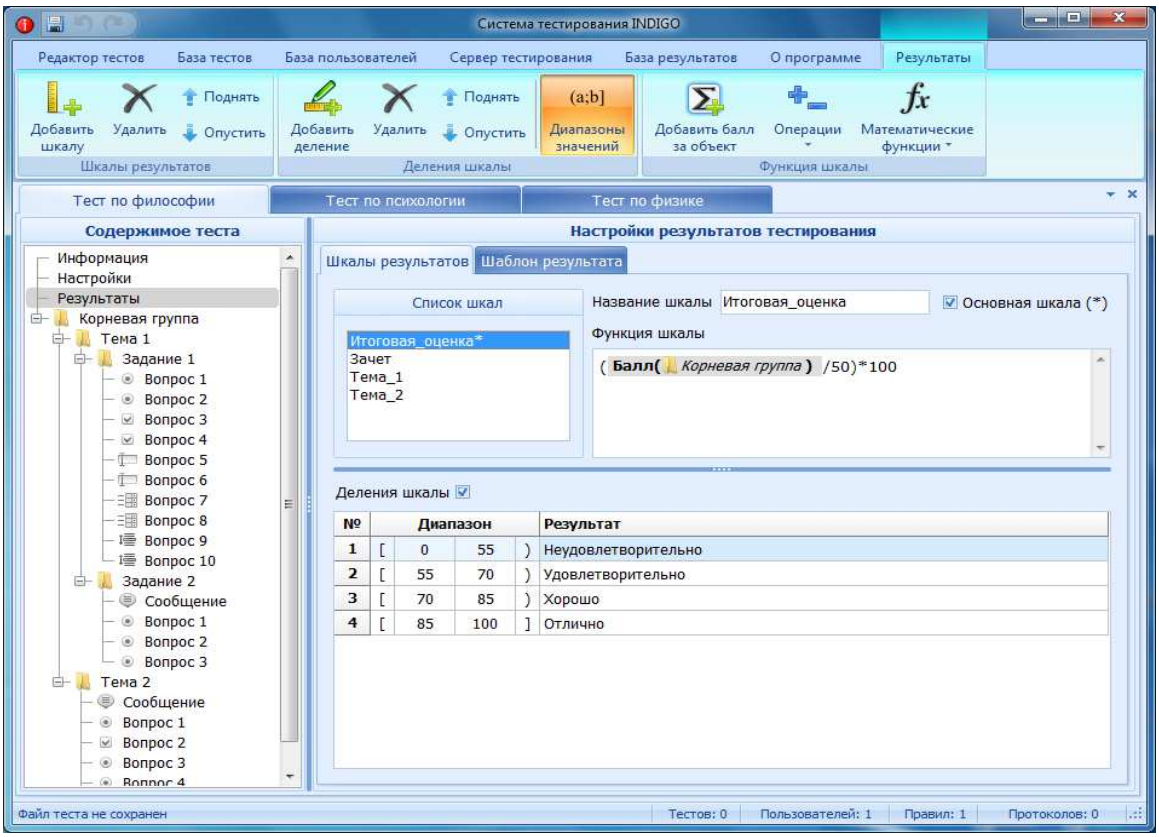


Рисунок 1 — Редактор шкал результатов системы тестирования INDIGO

К основным преимуществам можно отнести:

– доступный интерфейс пользователя;

– продукт совместим со всеми ОС семейства Windows (XP/2003/Vista /7/8);

– поддержка всех распространенных браузеров;

– широкие возможности конструктора тестов.

В свою очередь система имеет несколько недостатков:

– система является платной;

– отсутствие разделения администратор-преподаватель (преподаватель не всегда имеет навыки работы с подобного рода системами);

– относительно большой объем потребляемой памяти;

– высокие требования к оборудованию.

Также, «INDIGO» является ничем иным, как стандартным электронным тестом «вопрос-ответ». Элементы вариабельности и возможность создания активного тестирования отсутствует[14].

1.2.2 Система тестирования СИнТеЗ

Система предназначена для создания и редактирования тестов, проведения тестирований и анализа результатов. Функционал позволяет разбивать тесты на темы, указывать уровень сложности вопросов, составлять сценарии тестирования, т.е. указывать, сколько вопросов, по какой теме, какого уровня сложности будет задано обучаемым из всей базы вопросов. Имеет гибкие настройки времени, а именно, три временных режима:

– тестирование проводится без учета времени;

– время проведения тестирования ограничивается;

– время вычисляется, в зависимости от числа и сложности задаваемых вопросов.

Предусмотрено два способа определение порядка ответов на вопросы:

– обучаемый отвечает на вопросы в произвольном порядке и может изменить свой ответ;

– обучаемый отвечает на вопросы в заданном порядке и не может изменить свой ответ.

Система позволяет указывать цену вопросов в баллах, которые получит обучаемый за правильный ответ на вопрос того или иного уровня сложности (за более сложный ответ начисляется больше баллов). Задаются настройки перемешивания тем, вопросов и ответов:

– нужно ли перемешивать темы;

– нужно ли перемешивать вопросы;

– нужно ли перемешивать варианты ответов.

Возможны следующие варианты отчетов: общий отчет по группе или подробный отчет по каждому обучаемому.

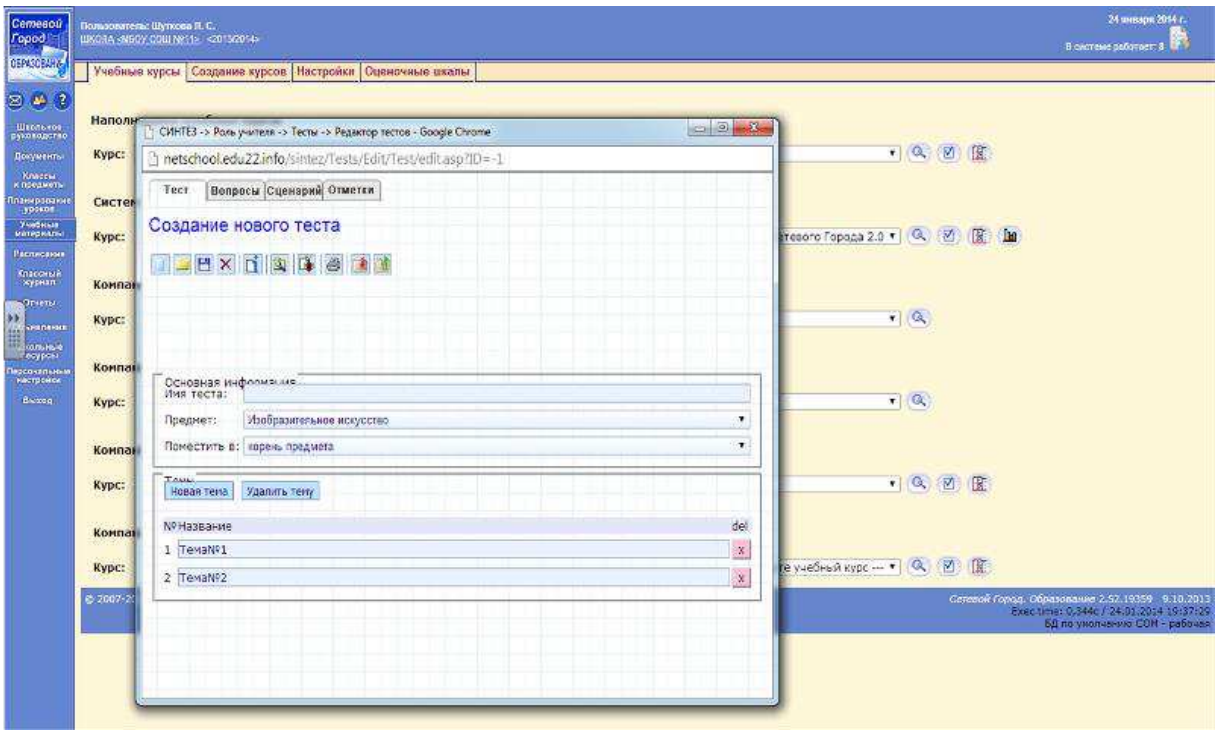


Рисунок 2— Редактор тестов системы тестирования СИнТеЗ

Данная система тестирования обладает неплохим функционалом, но также обладает серьёзными недостатками:

– в системе отсутствует возможность добавления в тест формул, что в свою очередь является важным критерием для создания тестирования по электростатике, в которой формулы — одна из важнейших и неотделимых частей;

– варианты написания и ввода ответа на вопрос могут быть различны, а система позволяет ввести лишь один вариант;

– ход решения система проверить не может.

1.2.3 Система тестирования UniTest System

UniTest System – полноценное решение для создания компьютерных тестов, проведения тестирования (как локально, так и по сети), детального анализа результатов тестирований и составления отчетов на предприятии или образовательном учреждении[6].

Пакет программ поддерживает все основные и множество дополнительных типов вопросов:

– простые вопросы на множественный выбор;

– вопросы с различными весами ответов;

– вопросы на выбор множества из элементов;

– упорядочивание последовательностей, установление соответствия;

– наличие/отсутствие ключевых слов в тексте ответа, прямой ввод.

В тестах можно применять произвольное форматирование для текстов, любую графику, таблицы, OLE-объекты, аудио и видео. Пакет может работать в интеграции с пакетом Microsoft Office, существуют возможности индивидуальной настройки тестов:

– работа с разделами тестов;

– создание динамически формируемых из базы вопросов тестов;

– режим обучения;

– система защиты от несанкционированного доступа и копирования тестов;

– создание автономных тестов;

– гибкие настройки оценивания.

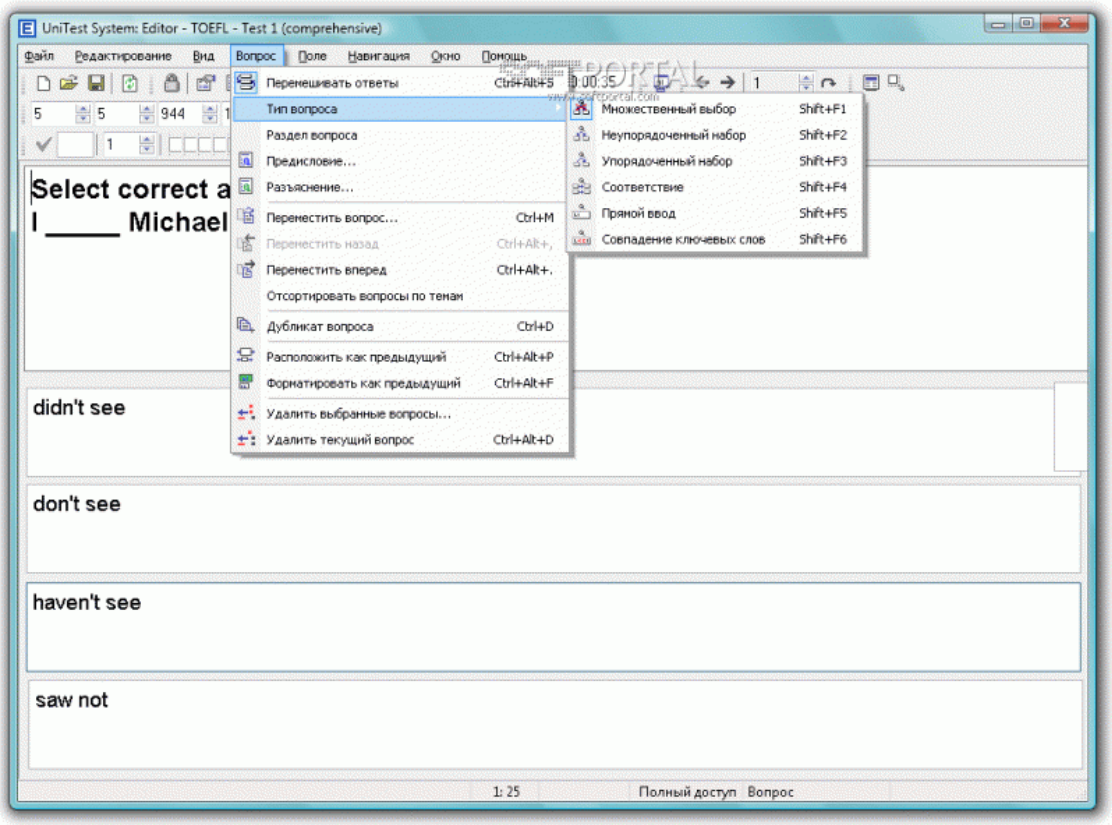


Рисунок 3 — Интерфейс системы UniTest System.

Являясь бесплатной, удобной, хорошо защищённой системой с возможностью локального и онлайн тестирования, система обладает следующими недостатками:

– плохая совместимость с последними версиями ОС Windows;

– высокие требования к техническим требованиям.

1.2.4 ЭОК Moodle

Система дистанционного обучения (СДО) Moodle широко известна в мире, и используется более чем в 100 странах[7]. По уровню предоставляемых возможностей СДО Moodle выдерживает сравнение с известными коммерческими СДО, в то же время выгодно отличается от них тем, что распространяется в открытом исходном коде — это дает возможность адаптировать систему под особенности конкретного образовательного проекта, а при необходимости

и встроить в нее новые модули. Система на данный момент является самой распространенной системой дистанционного обучения с самым большим количеством пользователей и разработчиков. Интерфейс системы дистанционного обучения Moodle(рисунок) переведен на 82 языка и используется почти в 50 тысячах организаций, из более чем 200 стран.

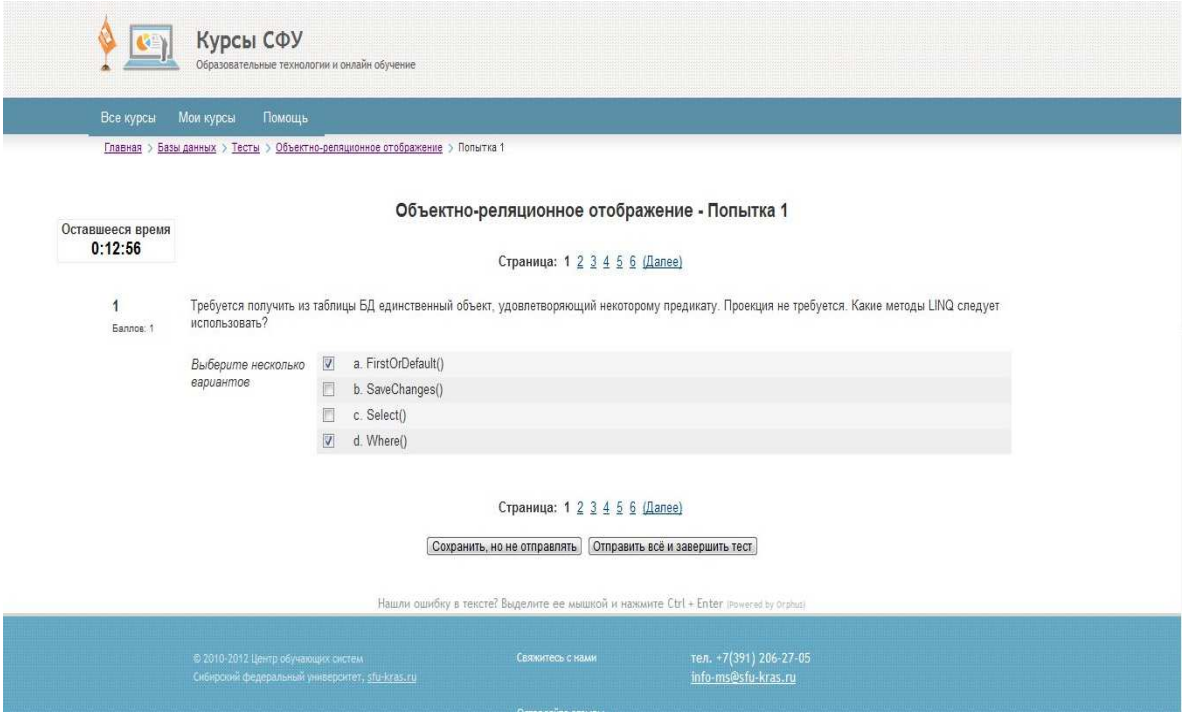


Рисунок 4 — Система Moodle на примере курсов СФУ.

Система обладает рядом достоинств:

– широкие коммуникационные возможности — одна из самых сильных сторон Moodle. Система поддерживает обмен файлами любых форматов — как между преподавателем и студентом, так и между самими студентами. Сервис рассылки позволяет оперативно информировать всех пользователей курса или отдельные группы о текущих событиях. Форум дает возможность организовать учебное обсуждение проблем, при этом обсуждение можно проводить по группам. К сообщениям в форуме можно прикреплять файлы любых форматов. Есть функция оценки сообщений — как преподавателями, так и студентами. Чат позволяет организовать обсуждение учебных проблем в режиме реального времени.

Сервисы «Обмен сообщениями», «Комментарий» предназначены для индивидуальной коммуникации преподавателя и студента: рецензирования работ, обсуждения индивидуальных учебных проблем;

– система создает и хранит портфолио каждого студента: все сданные им работы, все оценки и комментарии преподавателя к работам, все сообщения в форуме;

– преподаватель может создавать и использовать в рамках курса любую систему оценивания. Все отметки по каждому курсу хранятся в сводной ведомости.

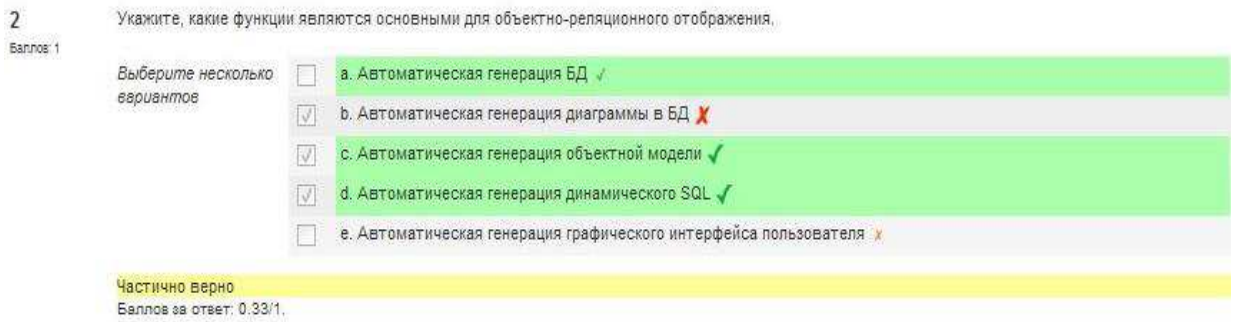


Рисунок 5 — Пример результатов теста

Варьируя сочетания различных элементов курса, преподаватель организует изучение материала таким образом, чтобы формы обучения соответствовали целям и задачам конкретных занятий.

К недостаткам системы можно отнести:

– стандартные дизайн решения системы;

– отсутствие профессиональной технической поддержки;

– необходимость собирать систему «с нуля»;

– сложность системы;

– требует технических компетенций в области веб-разработки от преподавателя.

Для современных учебных заведений необходим программный продукт, удовлетворяющий требованиям в полной мере. В идеальном случае, это должен быть комплекс модулей с возможностью выбора необходимых для учебного

процесса. Однако более важную роль представляет собой возможность подключения модулей, созданных на базе учебного заведения.

Система Moodle является хорошим выбором для высших учебных заведений, обладающих большим количеством институтов разного направления подготовки, особенно технической направленности. Однако проблемой подобных систем является то, что они создаются для широкого спектра задач, и попытка создать тест, к примеру, по электротехнике, будет представлять собой создание традиционного тестирования (система заданий, предъявляемая в порядке увеличения сложности в одно и то же время). Для создания тестов по редактированию и созданию простых электронных схем, или проверки правильности сборки необходимо создания особого модуля, способного:

– создать примитивы радиодеталей, из которых впоследствии будет создаваться тест;

– создать сами задания, а конкретно шаблон схемы которая является правильным ответом, количество элементов необходимых для сборки, варианты ответов, и т.д.;

– предоставить среду, в которой тестируемый будет составлять ответ на задачу;

– с помощью алгоритмов определить правильность предоставленных студентом ответов.

Подобные системы являются системами активного тестирования, и предлагают студенту нестандартную форму тестирования знаний. Активными являются тесты, которые предполагают диалог между тестируемым и тестом, возможность неоднозначных ответов, оценку степени освоения материала. Данные системы пока редкость для массового обучения, однако, подобные системы намного упростят учебный процесс для студентов технических специальностей. Данную систему, возможно заменить установленным преподавателем заданием, которое выполняется в аудитории в среде разработки, подобной NI Multisim, однако это требует индивидуального подхода к каждому студенту, тем самым ограничивая время и возможности преподавателя. Также к недо-

статкам данного подхода можно причислить высокую сложность освоения подобной среды разработки, стоимость использования и излишнюю функциональность.

1.3 Требования к приложению

Так как приложение является составляющей для целой системы, к нему существует ряд требований:

– общий формат хранения данных;

– общий способ представления данных для парсинга.

Для работы системы активного тестирования необходим редактор примитивов, в котором преподаватель может создать набор элементов с собственными характеристиками. Этот модуль будет доступен только преподавателю, и будет «сборщиком» элементов для создания тестов.

Работа данного модуля будет включать в себя создание необходимых для теста элементов и внедрение их в список деталей для постройки схемы. Создание элемента будет включать в себя:

1 Создание и наименование элемента из готовых графических примитивов, таких как круг, линия, треугольник и т.д.;

2 Задание базовых характеристик элемента;

3 Создание необходимых коммуникационных связей для соединения с другими элементами;

4 По окончанию работы, будет создан XML файл, в котором будет описан сам элемент как набор примитивов, характеристик, и способов коммуникации;

5 Сохранение элемента (запись в базу) для дальнейшей распаковки и использования в модуле сборщика.

Разрабатываемая система в целом, поможет значительно упростить учебный процесс, а также выявить пробелы в знаниях студентов. Подобные системы в будущем имеют все шансы вытеснить существующие традиционные виды те-

стирования, а также, развить саму идею, и развить её «активную» составляющую.

1.4 Инструменты разработки

1.4.1 Java

Java представляет собой объектно-ориентированный язык программирования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 году[8]. Программы на Java могут быть транслированы в байт-код, выполняемый на виртуальной Java-машине. Виртуальная машина это программа, обрабатывающая байт-код и передающая инструкции оборудованию, как интерпретатор, но с тем отличием, что байт-код, в отличие от текста, обрабатывается значительно быстрее. Тем самым достигается универсальность и адаптивность языка для любой платформы.

Значительным преимуществом языка является богатая библиотека. С одной стороны громадность библиотеки является одной из преград к изучению языка, и как следствие высокий порог вхождения. Но благодаря высокому порогу вхождения, популярность языка понизилась, в то время как востребованность языка осталась прежней. Богатая библиотека сводит тривиальную часть работы программиста к поиску готового решения. Что освобождает программиста от рутины и предоставляет ему возможность более серьезно вложиться в творческий аспект, сэкономить время и силы.

Язык Java, несмотря на различные способы оптимизации, всё же довольно ресурсоёмок и медлителен. Причины в следующем:

– автосборка мусора;

– компиляция "на лету" (Just In Time compilation);

– отказ от таких опасных механизмов как: арифметика указателей, неявное преобразование типов с потерей точности, функции первого класса.

Но это всё делает язык более независимым и безопасным. Автосборка мусора с одной стороны освобождает программиста от заботы освобождения памяти.

1.4.2 Swing

Swing — библиотека для создания графического интерфейса для программ на языке Java. Swing был разработан компанией Sun Microsystems. Он содержит ряд графических компонентов (англ. Swing widgets), таких как кнопки, поля ввода, таблицы и т. д.

Swing относится к библиотеке классов JFC, которая представляет собой набор библиотек для разработки графических оболочек. К этим библиотекам относятся Java 2D, Accessibility-API, Drag & Drop-API и AWT.

По сравнению с более ранней библиотекой AWT предоставляет более широкие возможности в области создания формы элементов, а также обладает кросс-платформенностью (а именно, компоненты имеют один и тот же дизайн для любой платформы, что позволяет делать форму компонента любой).

Компоненты Swing поддерживают специфические динамически подключаемые виды и поведения (англ. plugable look-and-feel), благодаря которому возможна адаптация к графическому интерфейсу платформы (то есть к компоненту можно динамически подключить другой, специфический для операционной системы, в том числе и созданный программистом вид и поведение).

Библиотека считается «Lightweight» (англ. Облегчённый). Это означает, что компоненты Swing отрисовываются на поверхности родительского окна, без использования компонентов операционной системы. В отличие от «Тяжелых» компонентов AWT, в приложении Swing может иметься только одно окно, и все прочие компоненты отрисовываются на ближайшем родителе, имеющем собственное окно (например, на JFrame).

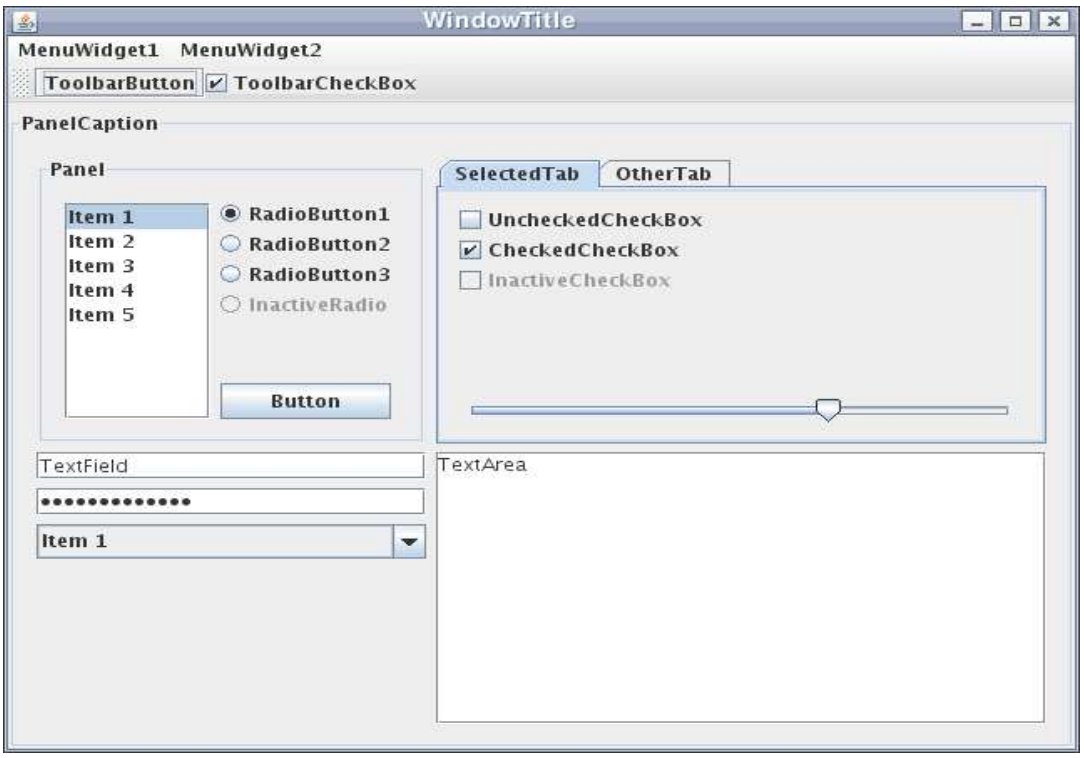


Рисунок 6 — Пример и внешний вид виджетов Java Swing

1.4.3 XML

XML (Extensible Markup Language) — язык разметки документов, позволяющий структурировать информацию разного типа, используя для этого произвольный набор инструкций. Сегодня XML может использоваться в любых приложениях, которым нужна структурированная информация — от сложных геоинформационных систем, с гигантскими объемами передаваемой информации до обычных программ, использующих этот язык для описания служебной информации. Можно выделить множество задач, связанных с созданием и обработкой структурированной информации, для решения которых может использоваться XML:

– эта технология может оказаться полезной для разработчиков сложных информационных систем, с большим количеством приложений, связанных потоками информации самой различной структурой;

– XML-документы могут использоваться в качестве промежуточного формата данных в трехзвенных системах. Обычно схема взаимодействия между серверами приложений и баз данных зависит от конкретной СУБД и диалекта SQL, используемого для доступа к данным. Если же результаты запроса будут представлены в некотором универсальном текстовом формате, то звено СУБД, как таковое, станет "прозрачным" для приложения;

– использование стилевых таблиц позволяет обеспечить независимое от конкретного устройства вывода отображение XML-документов;

– XML может использоваться в обычных приложениях для хранения и обработки структурированных данных в едином формате.

2 Разработка системы графических элементов

2.1 Составляющие системы

Система на данный момент представляет собой набор из трёх модулей, необходимых для создания электронного тестирования, на примере предмета «электротехника и схемотехника»:

1 редактор графических элементов ElCreator — цель данной магистерской диссертации. Предполагается, что это универсальное средство для создания элементов, необходимых для манипуляции ими в редакторе тестов. Включает в себя набор графических примитивов, а также средства для манипуляции ими. По результату работы модуля, будет получен файл формата .xml, в котором будет находиться список примитивов, и дополнительные свойства. Данный файл при открытии в редакторе теста преобразуется в графический элемент, необходимый для создания теста;

2 система хранения, передачи и проверки корректности заданий. Данный модуль представляет собой web-службу, предоставляющую доступ к информации об электротехнических схемах, представленных в виде XML-документа, которая собирается внешним приложением, и накапливается в базе данных. Далее посредством веб-сервиса, данная информация передается сторонним приложениям для отображения в удобном для них виде;

3 система ECEditorFX — среда для выполнения тестирования. Модуль использует службу хранения данных и подсистему формирования графических элементов. Программа-редактор тестов предоставляет базовой функционал: редактирование элементов графического образа задания (добавление, удаление, установка или изменение параметров элементов, установка связей между элементами и редактирование этих связей, предоставление простого доступа к базовым элементам), и функционал, решающий поставленные задачи (преобразо-

вание графического образа задания и его элементной базы в формат удобный для дальнейшего использования).

2.2 Формат конечных данных

2.2.1 Описание графического элемента

По результату работы программы, из набора элементов создаётся новый элемент, который в свою очередь является составляющей для подсистемы формирования графических образов. Элемент создаётся путём объединения списка примитивов в единый файл разметки.

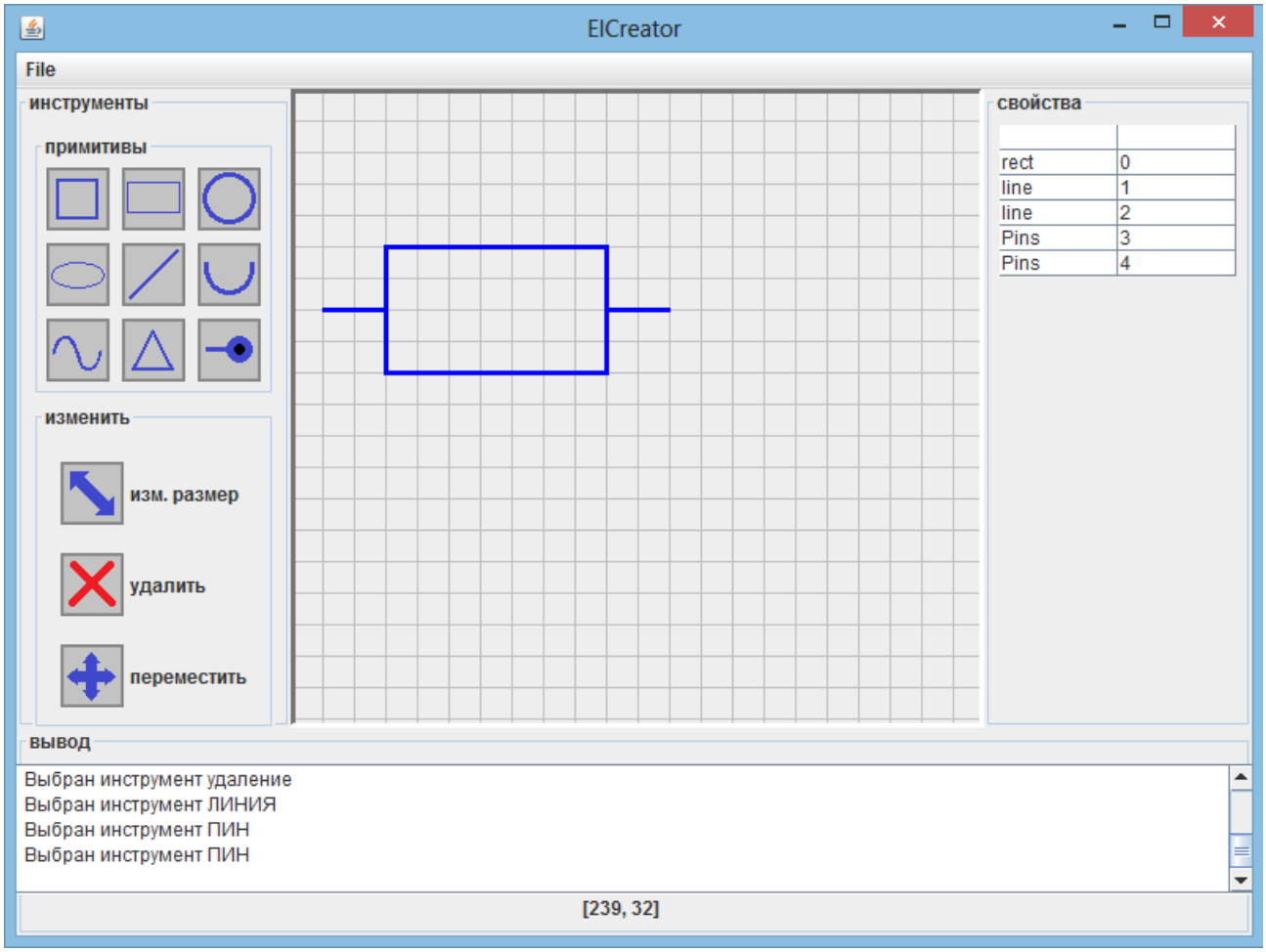


Рисунок 7 — Пример элемента состоящего из 3 примитивов и 2 пинов

Для данного элемента будет сохранен следующий XML-документ:

```
<el:element xmlns:el="http://www.example.org/element" xmlns:gp="ht
tp://www.example.org/graphicsPrimitives" xmlns:ut="http://www.exam
ple.org/util"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
in-
stance" id="r1" name="resistor" xsi:schemaLocation="http://www.exa
mple.org/element ../xsd/element.xsd ">
     <el:subClass subClassName="RESISTORS" subClassID="7"/>
     <el:type typeName="BASE_RESISTOR" typeID="2"/>
     <el:center x="130" y="140"/>
     <el:view>
          <gp:g>
               <gp:line x1="20" y1="140" x2="60" y2="140"/>
               <gp:line x1="200" y1="140" x2="240" y2="140"/>
               <gp:rect height="80" width="140" x="60" y="100"/>
          </gp:g>
     </el:view>
     <el:pins>
          <el:pin id="r1.p1" type="IN" visible="true">
               <ut:center x="0" y="10"/>
          </el:pin>
          <el:pin id="r1.p2" type="OUT" visible="true">
               <ut:center x="80" y="10"/>
          </el:pin>
     </el:pins>
</el:element>
```

Документ, описывающий графический элемент, содержит информацию о связях, количеству графических примитивов и их свойства.

2.2.2 Описание графических примитивов

Основная суть данной подсистемы — перевод графической информации представленной в виде массива данных в формат файла структурированных данных. Пользователь оперирует данными, используя графический интерфейс, соответственно информация представлена в виде геометрических фигур.

В качестве базового набора графических примитивов были выбраны:

– Линия;

– Квадратичная кривая Безье;

– Кубическая кривая Безье;

– Прямоугольник;

– Эллипс.

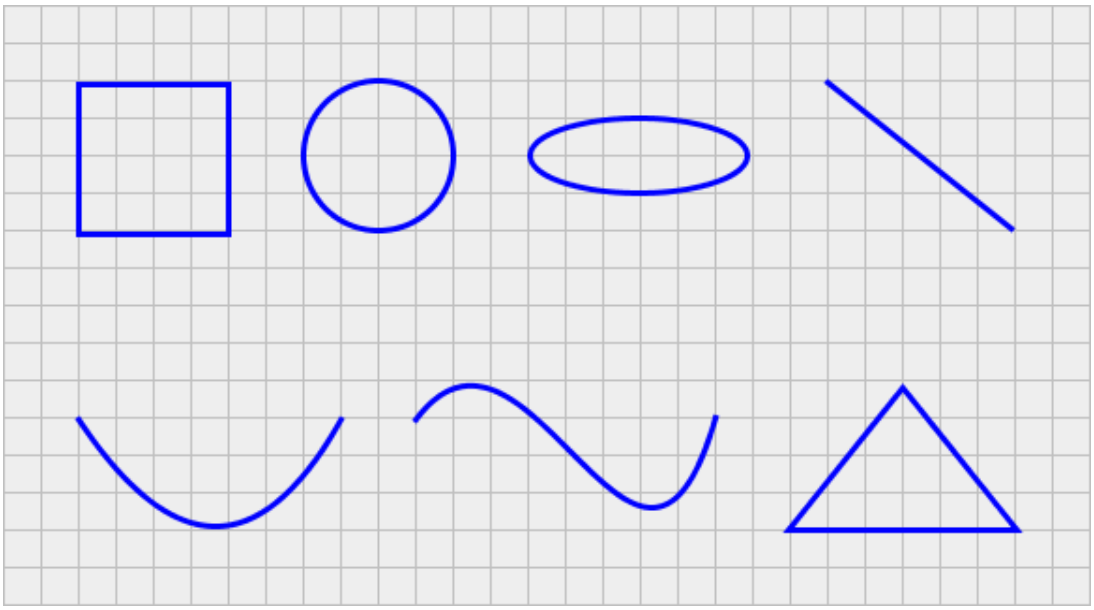


Рисунок 8 — Графические примитивы, представленные в редакторе

Все эти примитивы входят в интерфейс `java.awt.Shape`. Соответственно, в качестве массива данных используется массив `Shape [] shapes`:

```
Shape [] shapes = new Shape[MAX];//инициализация массива графических примитивов
```

Переменная `int MAX` определяет максимальное количество примитивов. Для подсчёта количества записанных примитивов используется переменная `int numOfShapes`. Для записи в массив используются классы:

Прямоугольник

`Java.awt.Rectangle` — используется для создания квадрата и прямоугольника, например:

```
shapes[numOfShapes] = new Rectangle(x, y, width, width);
```

Описание атрибутов:

x, y – координаты левого верхнего угла прямоугольника.

height, width — высота и ширина прямоугольника.

Эллипс

`Java.awt.geom.Ellipse2D.Double` — Используется для создания круга и овала.

```
shapes[numOfShapes] = new Ellipse2D.Float(x, y, width, width);
```

Описание атрибутов:

x, y – координаты левого верхнего угла прямоугольника описывающего эллипс.

height, width — высота и ширина эллипса.

Линия

`java.awt.geom.Line2D` — Используется для создания линии, а также, для создание пинов (начальная и конечная точка являются входом и выходом соответственно). Центр линии используется для определения центра элемента.

```
shapes[numOfShapes] = new Line2D.Float(startX, startY, endX, endY);
```

Описание атрибутов:

startX, startY — x, y координаты точки начала прямой.

endX, endY — x, y координаты точки конца прямой.

Квадратичная линия Безье

`java.awt.geom.QuadCurve2D` — используется для создания квадратичной линии.

```
shapes[numOfShapes] = new QuadCurve2D.Float(startX, startY, ctrlx, ctrly, endX, endY);
```

Описание атрибутов:

startX, startY — x, y координаты начальной точки кривой.

ctrlx, ctrly — x, y координаты контрольной точки кривой.

endX, endY — x, y координаты конечной точки кривой.

Кубическая кривая Безье

java.awt.geom.CubicCurve2D — используется для создания кубической кривой.

```
shapes[numOfShapes] = new CubicCurve2D.Float(startX, startY,
ctrlx, ctrly, ctrlx2, ctrly2, endX, endY);
```

Описание атрибутов:

startX, startY — x, y координаты начальной точки кривой.

ctrlx, ctrly — x, y координаты первой контрольной точки кривой.

ctrlx2, ctrly2 — x, y координаты второй контрольной точки кривой.

endX, endY — x, y координаты конечной точки кривой.

Полигон

`java.awt.Polygon` — Используется для создания треугольника.

```
shapes[numOfShapes] = new Polygon(TriangleArrayX, TriangleArrayY,
3);
```

Описание атрибутов:

TriangleArrayX — массив, содержащий координаты по оси X для построения полигона.

TriangleArrayY — массив, содержащий координаты по оси Y для построения полигона.

“3” — Количество углов, по которым будет построен полигон.

2.2.3 Правила описания графических элементов

Созданный элемент представляет собой XML-документ, в котором описаны примитивы, их свойства, и тип связи с элементами (пины и их типы). Также внутри документа описана идентификационная информация, такая как

тип элемента, его класс и его ID. Данный XML-документ будет использован приложении ECEditorFX, где описаны правила их структуры в документах .xsd.

Пример элемента, полученного по результату работы программы:

```
<el:element  id="r555" name="test"…>
     <el:subClass subClassID="7" subClassName="ELEMENTS"/>
     <el:type typeID="2" typeName="TEST_ELEMENT"/>
     <el:center x="240.0" y="180.0"/>
     <el:view>
          <gp:g>
               <gp:rect height="116" width="201" x="141" y="121"/>
               <gp:line x1="240.0" x2="240.0" y1="40.0" y2="320.0"/>
          </gp:g>
     </el:view>
     <el:pins>
          <el:pin id="i1.p0" type="IN" visible="true">
               <ut:center x="240" y="41"/>
          </el:pin>
          <el:pin id="i1.p0" type="OUT" visible="true">
               <ut:center x="240" y="320"/>
          </el:pin>
     </el:pins>
</el:element>
```

Данный XML документ характерен для элемента типа резистор. На рисунке 9 указаны соответствующие переменные, описанные в документе.

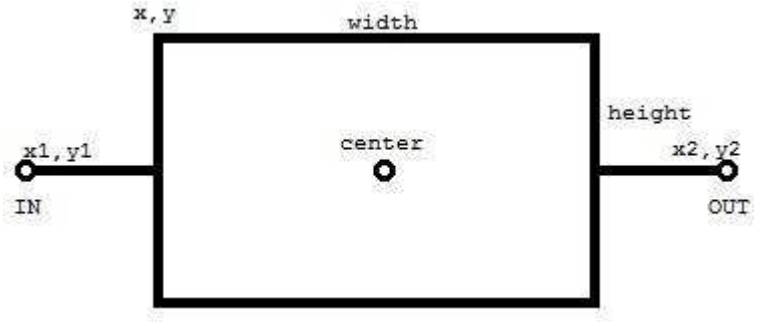


Рисунок 9 — Указание переменных для элемента

Теги описаны в правилах описаний графических элементов, содержащихся в приложении В.

2.2.4 Обмен информацией между системами

Система графических элементов взаимодействует с сервисом путем хранения графических элементов с возможностью загрузки и выгрузки элементов для последующей обработки.
Взаимодействие с редактором элементов происходит путем подключения к сервису с помощью HTTP-клиента, в частности запроса POST. Исходя из архитектуры сервиса REST, следовательно, загрузка происходит небольшими запросами по одному элементу за один запрос.

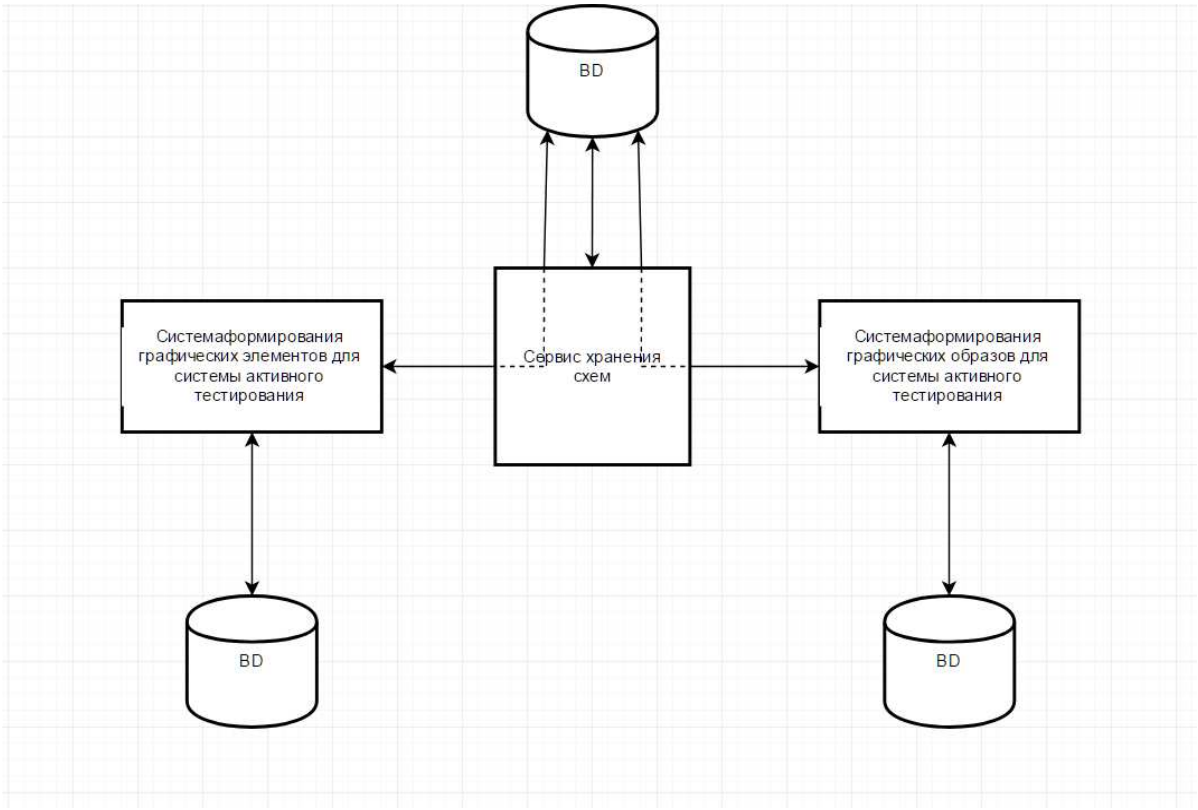


Рисунок 10 — Схема обмена информацией между системами

2.3 Графический интерфейс

2.3.1 Общий вид

Графический интерфейс построен на основе классической библиотеки Swing. Библиотека Swing является универсальным средством для создания кроссплатформенного интерфейса, что позволяет запустить данное приложение на различных системах. Главное окно программы делится на несколько участков (Рисунок 10). Кнопки, использующиеся на `JPanel primitives`, и на `JPanel tools` являются элементами `JRadioButton`, внешний вид которых изменён на `ImageIcon`. Это позволяет выбрать единственный элемент при создании примитива на рабочей области. Иконки загружаются из указанной директории, в массив `icon[]`, а затем в функции `addIcon()` они присваиваются соответствующему `JRadioButton`.

```
for (int i = 0; i<=23; i++)
{
     Icon1[i] = new ImageIcon(url+i+".png");
}
addIcon();
```

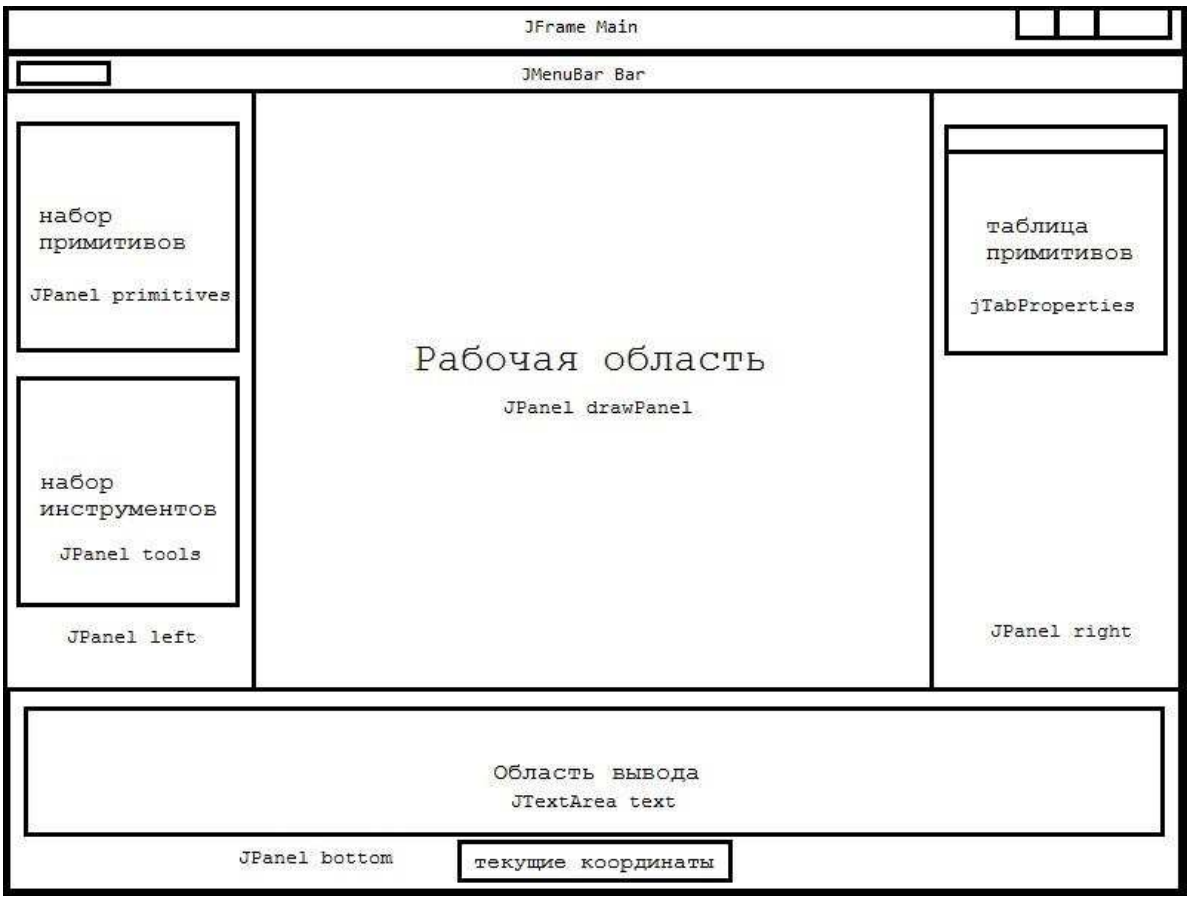


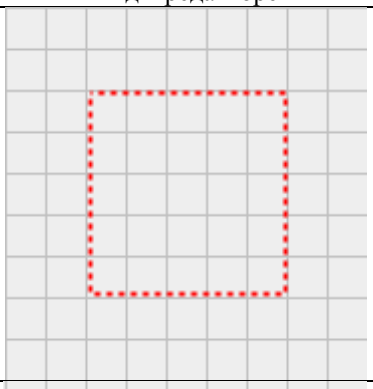
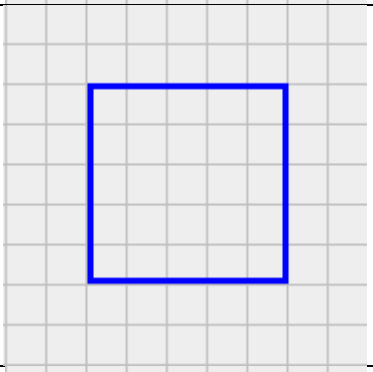
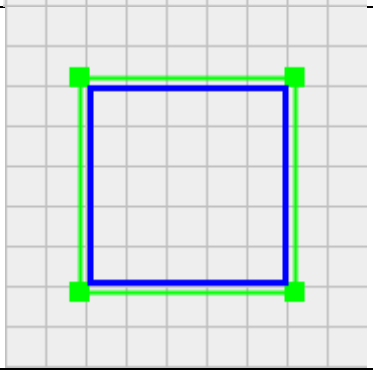
Рисунок 11 — Общая схема окна приложения с указанием панелей

Текстовая панель `JTextArea text` является консолью, которая указывает на действия, произведённые во время работы программы. `jTabProperties` является таблицей, содержащей `DefaultTableModel dtm`, который содержит информацию об элементах, добавленных на рабочую область. При нажатии на строку с определённым элементом, данный элемент выбирается из массива, и выделяется зелёным фоном, готовый для изменения характеристик или переноса.

2.3.2 Действия над примитивами

Внешний вид примитива зависит от его состояния (создание, выделение, изменение размера, перенос). Внешний вид состояний элементов, и их значения представлены в таблице 1.

Таблица 1 — Отображение состояний примитивов

Вид в редакторе	Действие	Описание
	Создание, перемещение.	Создаваемый или перемещаемый примитив отображается пунктирной линией.
	Готовность	Готовый элемент отображается синей линией, что означает него неактивность.
	Выделение, изменение размера.	Выделяемый элемент обрисовывается дополнительным большим Rectangle зелёного цвета, с четырьмя дополнительными малыми. Правый нижний используется для изменения размера графического примитива.

Для манипуляции над элементом, вначале присваивается значение переменной `int currentShapeIndex`, которое, соответственно, является идентификатором конкретного примитива. Поиск примитива выглядит следующим образом:

```
public int getShape(int x, int y) {//получение shape из массива,
подпадающего под координаты
int sensorX = x - 4 / 2;//создание «сенсора»
int sensorY = y - 4 / 2;
int sensorwidth = 4;
int sensorheight = 4;
for (int i = 0; i < numOfShapes; i++)
{
```

```
        if (shapes[i].contains(x, y))//если координаты соответствуют
shape, то возвращается его id
        {
            return i;
        }
        else if (shapes[i].intersects(sensorX, sensorY, sensorwidth,
        sensorheight))
        {
            return i;
        }
}
return -1;
}
```

Если для поиска обычного примитива достаточно «найти» его координаты с помощью координат курсора, то для Line2D требуется создать «сенсор», так как всё, чем он обладает в качестве свойств — это координаты начальной и конечной точек. Для создания «сенсора» используется метод `Shape.intersects()`, позволяющий создать квадрат, при помощи которого можно проверить пересекается ли он с линией.

Действия над элементом происходят исходя из комбинаций значений контрольных переменных. Список всех контрольных переменных представлен в Таблице 2.

Таблица 2 — Контрольные переменные и их значения

Переменная	Значение
`int drawstate`	Состояние отрисовки, если 0, то примитив отрисовывается без записи в массив.
`int inside`	Если 1, то курсор назодится внутри переменной
`int checkshape`	Значение переменной задаётся по результату функции `getShape()`. Значение от 0 до n типов примитивов.
`int startresize`	Если 1, то возможно изменение размеров примитива используя `mouseDragged(MouseEvent e)`
`int lastSelected`	В переменной содержится id последнего выделенного элемента. Значение от 0 до n типов примитивов.
`int select`	Если 1, значит конкретный элемент был выделен.

ЗАКЛЮЧЕНИЕ

В процессе выполнения работы разработана подсистема формирования графических образов для системы активного тестирования. Разработанная система реализована в виде приложения с использованием языка программирования Java и библиотек Swing, AWT.

Разработанная подсистема создаёт графические элементы, и сохраняет их в документе XML. В формате, удобном для дальнейшего использования в подсистеме формирования графических образов.

СПИСОК СОКРАЩЕНИЙ

ЭОК — Электронные образовательные курсы

НУЛ САПР — Научно-учебная лаборатория систем автоматизированного проектирования

AWT — Abstract Window Toolkit

XML — Extensible Markup Language

XLS — Extensible Stylesheet Language

СДО — Система дистанционного обучения

NI Multisim — National instruments Multisim

JFC — Java Foundation Classes

API — Application programming interface

OLE — Object Linking and Embedding

СИнТеЗ — Система информационного тестирования знаний

ОС — Операционная система

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Публикации автора:

1. Кацунова, А. С. Тесты как элемент контура автоматического управления информационным процессом обучения / С. А. Бронов, А. С. Кацунова, И. К. Камилов, Б. Р. Хождаев, Д. С. Рогов, Д. С. Степанов, Д. М. Севостьянов, Е. О. Мартыненко // Информатизация образования и методика электронного обучения : материалы I Международной научной конференции в рамках IV Международного научно-образовательного форума «Человек, семья и общество: история и перспективы развития» (Красноярск, 27–30 сентября 2016 г.) // под общ. ред. М. В. Носкова. – Красноярск : Сиб. федер. ун-т, 2016. – С. 56–60. – ISBN 978-5-7638-3559-5.

2. Кацунова, А. С. Активная система тестирования графического материала: подсистема формирования элементов / С. А. Бронов, А. С. Кацунова, И. К. Камилов, Д. М. Севостьянов // Информатизация образования и методика электронного обучения : материалы I Международной научной конференции в рамках IV Международного научно-образовательного форума «Человек, семья и общество: история и перспективы развития» (Красноярск, 27–30 сентября 2016 г.) // под общ. ред. М. В. Носкова. – Красноярск : Сиб. федер. ун-т, 2016. – С. 52–56. – ISBN 978-5-7638-3559-5.

Использованные источники:

1. Электронный учебный курс [Электронный ресурс]. — Режим доступа: https://ru.wikipedia.org/wiki/Электронный_учебный_курс

2. Мультимедиа-курсы: методология и технология разработки [Электронный ресурс]. — https://ido.tsu.ru/files/pub2002/7_2002_vum_dem_mozh.pdf

3. Зайнутдинова, Л. Х. Создание и применение электронных учебников (на примере общетехнических дисциплин) / Л. Х. Зайнутдинова. — Астрахань: Изд-во ЦНТЭП, 1999.

4. Пастущак, Т. Н. Создание электронного курса : лекция в СДО moodle. учебно-методическое пособие / Т. Н. Пастущак , С. С. Соколов, А. А. Рябова. – Санкт-Петербург : СПГУВК, 2012. – 45 с.

5. Учебник о формате JSON [Электронный ресурс]. — Режим доступа: https://www.w3schools.com/js/js_json_intro.asp

6. Система тестирования INDIGO [Электронный ресурс]. — Режим доступа : http://indigotech.ru/

7. Система тестирования UniTest System [Электронный ресурс]. — Режим доступа : http://sight2k.com/rus/unitest

8. Система тестирования Moodle [Электронный ресурс]. — Режим доступа : https://moodle.org/?lang=ru

9. Java Documentation [Электронный ресурс]. — Режим доступа: https:// docs.oracle.com

10. Аванесов, В. С. Композиция тестовых заданий / В. С. Аванесов. — Москва : Центр тестирования, 2002.

11. Бронов, С. А. Автоматизированный анализ и синтез учебных планов вуза на основе массива дидактических единиц / С. А. Бронов, Е. А. Степанова, К. В. Калиновский, И. В. Соколов, Н. С. Храброва // Вестник КрасГАУ. — 2014.

12. Перова, Ю. П. Технологии тестирования в дистанционном обучении [Электронный ресурс]. / Ю.П. Петрова — Режим доступа : http://old.tusur.ru/filearchive/reports-magazine/2015-35-1/24.pdf

13. Среда разработки NI Multisim [Электронный ресурс]. — Режим доступа : http://russia.ni.com/multisim

14. Буланая, М. А. Информационные управляющие системы и компьютерный мониторинг (ІУС КМ — 2013) / М. А. Буланая, С. А. Цололо // Электронный архив ДонНТУ.— 2013.

15. Обзор релиза Moodle 3.0 Highlights [Электронный ресурс]. — Режим доступа: http://www.umass.edu/it/support/moodle/moodle-30-highlights.

16. Углев, В. А. Обучающее компьютерное тестирование / В. А. Углев. // Теоретические и прикладные вопросы современных информационных технологий Улан-Удэ : ВСГТУ, 2007. – 312 с.

17. Андреев, С. В. Обзор систем автоматизированного тестирования / С. В. Андреев, В. В. Найханов // Москва : Центр тестирования, 2002.

18. Введение в разработку графического интерфейса [Электронный ресурс]. — Режим доступа: https://netbeans.org/kb/docs/java/gui-functionality_ru.html

19. Степанов, Д. С. Подсистема формирования графических образов для системы активного тестирования / Д. С. Степанов. — Красноярск : ИКИТ СФУ, 2017.

20. Введение в язык Java [Электронный ресурс]. — Режим доступа: http://www.java-study.ru/2-vvedenie.html

ПРИЛОЖЕНИЕ А

Листинг записи примитивов в массив

```
public void paintComponent(Graphics g) {
Graphics2D g2 = (Graphics2D) g;
Graphics2D g3 = (Graphics2D) g;
paintBackground(g2);  // отрисовка сетки
float[] dashingPattern1 = {3f, 3f};
(Graphics2D) g).setStroke(new BasicStroke(2f, BasicStroke.CAP_BUTT,
BasicStroke.JOIN_MITER, 1.0f, dashingPattern1, 2.0f));
((Graphics2D) g).setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);
g.setColor(Color.RED);//красный цвет и пунктир для ри-сования
x = (startX < endX) ? startX : endX;
y = (startY < endY) ? startY : endY;
int width = endX - startX + 1;
if (width < 0) width = -width;
int height = endY - startY + 1;
if (height < 0) height = -height;
if(select == 1 && inside == 1 && dragged!=1 || table-Click != -
1){//отрисовка зелёной рамки
paintDots(g3);
}
if(dragged == 1 && currentShapeIndex >= 0 || star-tresize ==
1){//перерисовка при переносе
if(checkshape == 1 || checkshape == 2 )//прямоугольник
{
((Graphics2D) g).draw(temp);
}
if(checkshape == 3)//эллипс
{
g.drawOval(tempx, tempy, temph, temph);
}
if(checkshape == 4)//овал
{
g.drawOval(tempx, tempy, tempw, temph);
```

```
}
if(checkshape == 5 || checkshape == 9)//линия
{
g.drawLine(tempx, tempy, temph, tempw);
}
if(checkshape == 6)//кривая
{
((Graphics2D) g).draw(curvetemp);
}
if(checkshape == 7)
{
((Graphics2D) g).draw(qcurvetemp);
}

}
if (currentShapeIndex < 0 && numOfShapes < MAX) // создание(рисование
элемента)
{
if (drawmethod == 1){
      if (drawstate == 1 && inside == 0)//если мышь отпущена
      {
            shapes[numOfShapes] = new Rectangle(x, y, width, width);
            addProperties(g);
            addLayer(0, numOfShapes);
            drawmethod = 0;
      }
      g.drawRect(x, y, width, width);//рисовать, если кнопка мыши зажата
}
if (drawmethod == 2){
      if (drawstate == 1 && inside == 0)
      {
            shapes[numOfShapes] = new Rectangle(x, y, width, height);
            addProperties(g);
            addLayer(1, numOfShapes);
            drawmethod = 0;
            //temp = null;

      }
```

```
      g.drawRect(x, y, width, height);
}
if (drawmethod == 3){
      if (drawstate == 1 && inside == 0)
      {
            shapes[numOfShapes] = new El-lipse2D.Float(x, y, width,
width);
            addProperties(g);
            addLayer(2, numOfShapes);
            drawmethod = 0;
            //temp = null;
      }
      g.drawOval(x, y, width, width);
}
if (drawmethod == 4){
      if (drawstate == 1 && inside == 0)
      {
            shapes[numOfShapes] = new El-lipse2D.Float(x, y, width,
height);
            addProperties(g);
            addLayer(3, numOfShapes);
            drawmethod = 0;
            //temp = null;
      }
      g.drawOval(x, y, width, height);
}
if (drawmethod == 5){
      if (drawstate == 1 && inside == 0)
      {
            shapes[numOfShapes] = new Line2D.Float(startX, startY, endX,
endY);
            addProperties(g);
            addLayer(4, numOfShapes);
            drawmethod = 0;
            //temp3 = null;
      }
      g.drawLine(startX, startY, endX, endY);
}
```

```
if (drawmethod == 6){
      if(ctrlpoint == 0)
      {
            g.drawLine(startX, startY, endX, endY);
      }
      else
      {
            QuadCurve2D q = new QuadCurve2D.Float(startX, startY, ctrlx,
ctrly, endX, endY);
            if (drawstate == 1 && inside == 0 && ctrlpoint == 2)
            {
      shapes[numOfShapes] = curvetemp;
                  addProperties(g);
                  addLayer(5, numOf-Shapes);
                  drawmethod = 0;
                  ctrlpoint = 0;
            }
            ((Graphics2D) g).draw(q);
      }

}
if (drawmethod == 8){
      Polygon drawTriangle = new Poly-gon(TriangleArrayX, TriangleArrayY,
3);
      if (drawstate == 1 && inside == 0)
      {
            shapes[numOfShapes] = new Poly-gon(TriangleArrayX,
TriangleArrayY, 3);
            addProperties(g);
            addLayer(7, numOfShapes);
            drawmethod = 0;                                    }
      g.drawPolygon(drawTriangle);
}
if (drawmethod == 9){
      if (drawstate == 1 && inside == 0)
      {
shapes[numOfShapes] = new Line2D.Float(startX, startY, endX, endY);
pinlist[pincount][0] = Integer.toString(startX);
```

```
pinlist[pincount][1] = Integer.toString(startY);
pinlist[pincount][2] = "IN";
pinlist[pincount][3] = Integer.toString(endX);
pinlist[pincount][4] = Integer.toString(endY);
pinlist[pincount][5] = "OUT";
pinlist[pincount][6] = Integer.toString(numOfShapes);
pincount++;
     }
          centerpoint = new
Point2D.Float((startX+endX)/2,(startY+endY)/2);
          addProperties(g);
          addLayer(8, numOfShapes);
          drawmethod = 0;
     }
     g.drawLine(startX, startY, endX, endY);
     g.drawOval(startX-5, startY-5, 10, 10);
     g.drawOval(endX-5, endY-5, 10, 10);
}
if (drawmethod == 7){
     if(ctrlpoint == 0)
     {
          g.drawLine(startX, startY, endX, endY);
     }
     else
     {
          CubicCurve2D q = new Cubic-Curve2D.Float(startX, startY,
ctrlx, ctrly, ctrlx2, ctrly2, endX, endY);
          if (drawstate == 1 && inside == 0 && ctrlpoint == 5)
          {
               text.append(Double.toString(qcurvetemp.getX1())+"
"+Double.toString(qcurvetemp.getY1())+"  "+Integer.toString(ctrlx)+"
"+Integer.toString(ctrly)+"  "+Integer.toString(endX)+"
"+Integer.toString(endY)+"  "+"\n");
               shapes[numOfShapes] = qcurvetemp;
               addProperties(g);
               addLayer(6, numOf-Shapes);
               drawmethod = 0;
               ctrlpoint = 0;
```

```
            }
            ((Graphics2D) g).draw(q);
      }
}
if (drawmethod == 0){
      //
}
}
((Graphics2D) g).setStroke(new BasicStroke(3));//перерисовка всех элеме-
тов
((Graphics2D) g).setPaint(Color.BLUE);
for (int i = 0; i < numOfShapes; i++) {
g2.draw(shapes[i]);
}
}
```

ПРИЛОЖЕНИЕ Б

Листинг парсинга примитивов, и записи их в файл

```
public void save(){
try {
//text.append("количество примитивов всего: "+numOfShapes+"\n");
DocumentBuilderFactory docFactory = DocumentBuilderFac-
tory.newInstance();
DocumentBuilder docBuilder = docFacto-ry.newDocumentBuilder();
Document doc = docBuilder.newDocument();
Element root = doc.createElement("el:element");

doc.appendChild(root);
root.setAttribute("id", "r555");

root.setAttribute("name", "resistor");

root.setAttribute("xmlns:el", "http://www.example.org/element");
root.setAttribute("xmlns:gp",
"http://www.example.org/graphicsPrimitives");
root.setAttribute("xmlns:ut", "http://www.example.org/util");
root.setAttribute("xmlns:xsi", "http://www.w3.org/2001/XMLSchema-
instance" );

root.setAttribute("xsi:schemaLocation","http://www.example.org/element../
xsd/element.xsd");

Element subclass = doc.createElement("el:subClass");
subclass.setAttribute("subClassName", "RESISTORS");
subclass.setAttribute("subClassID", "7");
root.appendChild(subclass);

Element type = doc.createElement("el:type");
type.setAttribute("typeID", "2");
type.setAttribute("typeName", "BASE_RESISTOR");
root.appendChild(type);
```

```
Element center = doc.createElement("el:center");
root.appendChild(center);
center.setAttribute("x", Dou-ble.toString(centerpoint.getX())); //впели
центр,модифицируй пины в линию, координаты будут пинами, центр-серединой.
center.setAttribute("y", Dou-ble.toString(centerpoint.getY()));
root.appendChild(center);

Element view = doc.createElement("el:view");
root.appendChild(view);
Element gp = doc.createElement("gp:g");
view.appendChild(gp);

for(int i = 0; i<numOfShapes; i++){
if(typeOfShape[ShapeProperties[i]-1] != "Pins")
{
      Element part = doc.createElement("gp:"+
typeOfShape[ShapeProperties[i]-1]);
      if(typeOfShape[ShapeProperties[i]-1] == "rect")
{
      temp = shapes[i].getBounds();
      part.setAttribute("x", Inte-ger.toString(temp.x));
      part.setAttribute("y", Inte-ger.toString(temp.y));
      part.setAttribute("width", Inte-ger.toString(temp.width));
      part.setAttribute("height", Inte-ger.toString(temp.height));
}else if(typeOfShape[ShapeProperties[i]-1] == "el-lipse"){
      temp4 = (Ellipse2D.Float) shapes[i];
      part.setAttribute("x", Float.toString((float) temp4.getX()));
      part.setAttribute("y", Float.toString((float) temp4.getY()));
      part.setAttribute("width", Float.toString((float)
temp4.getWidth()));
      part.setAttribute("height", Float.toString((float)
temp4.getHeight()));
}else if(typeOfShape[ShapeProperties[i]-1] == "line"){
      temp3 = (Line2D.Float) shapes[i];
      part.setAttribute("x1", Float.toString((float) temp3.getX1()));
      part.setAttribute("y1", Float.toString((float) temp3.getY1()));
      part.setAttribute("x2", Float.toString((float) temp3.getX2()));
```

```
     part.setAttribute("y2", Float.toString((float) temp3.getY2()));
}else if(typeOfShape[ShapeProperties[i]-1] == "line"){
     temp3 = (Line2D.Float) shapes[i];
     part.setAttribute("x1", Float.toString((float) temp3.getX1()));
     part.setAttribute("y1", Float.toString((float) temp3.getY1()));
     part.setAttribute("x2", Float.toString((float) temp3.getX2()));
     part.setAttribute("y2", Float.toString((float) temp3.getY2()));
}else if(typeOfShape[ShapeProperties[i]-1] == "quadCurve"){
     curvetemp = (QuadCurve2D.Float) shapes[i];
     part.setAttribute("x1", Float.toString((float) curvetemp.getX1()));
     part.setAttribute("y1", Float.toString((float) curvetemp.getY1()));
     part.setAttribute("ctrlx", Float.toString((float)
curvetemp.getCtrlX()));
     part.setAttribute("ctrly", Float.toString((float)
curvetemp.getCtrlY()));
     part.setAttribute("x2", Float.toString((float) curvetemp.getX2()));
     part.setAttribute("y2", Float.toString((float) curvetemp.getY2()));
}else if(typeOfShape[ShapeProperties[i]-1] == "cu-bicCurve"){
     qcurvetemp = (CubicCurve2D.Float) shapes[i];
     part.setAttribute("x1", Float.toString((float) curvetemp.getX1()));
     part.setAttribute("y1", Float.toString((float) curvetemp.getY1()));
     part.setAttribute("ctrlx1", Float.toString((float)
qcurvetemp.getCtrlX1()));
     part.setAttribute("ctrly1", Float.toString((float)
qcurvetemp.getCtrlY1()));
     part.setAttribute("ctrlx2", Float.toString((float)
qcurvetemp.getCtrlX2()));
     part.setAttribute("ctrly2", Float.toString((float)
qcurvetemp.getCtrlY2()));
     part.setAttribute("x2", Float.toString((float) curvetemp.getX2()));
     part.setAttribute("y2", Float.toString((float) curvetemp.getY2()));
}else if(typeOfShape[ShapeProperties[i]-1] == "polygon"){
     temp2 = (Polygon) shapes[i];
     for(int a = 0; a<3; a++){
          points = points + (Inte-
ger.toString(temp2.xpoints[a])+","+Integer.toString(temp2.ypoints[a])+"
");
          part.setAttribute("points", points);
```

```
      }
}
gp.appendChild(part);
}
}
//выпили лишнее снизу
Element pins = doc.createElement("el:pins");
root.appendChild(pins);

for(int i = 0;i<pincount;i++)
{
Element pinin = doc.createElement("el:pin");
pinin.setAttribute("id", "i1.p"+i);
pinin.setAttribute("type", pinlist[i][2]);
pinin.setAttribute("visible", "true");
pins.appendChild(pinin);

Element pinincenter = doc.createElement("ut:center");
pinincenter.setAttribute("x", pinlist[i][0]);//замени пины, вместо точки
впили отдельные переменные
pinincenter.setAttribute("y", pinlist[i][1]);
pinin.appendChild(pinincenter);

Element pinout = doc.createElement("el:pin");
pinout.setAttribute("id", "i1.p"+i);
pinout.setAttribute("type", pinlist[i][5]);
pinout.setAttribute("visible", "true");
pins.appendChild(pinout);

Element pinoutcenter = doc.createElement("ut:center");
pinoutcenter.setAttribute("x", pinlist[i][3]);//замени пины, вместо точки
впили отдельные переменные
pinoutcenter.setAttribute("y", pinlist[i][4]);
pinout.appendChild(pinoutcenter);
}

TransformerFactory transformerFactory = TransformerFac-
tory.newInstance();
```

```
Transformer transformer = transformerFacto-ry.newTransformer();
DOMSource source = new DOMSource(doc);
StreamResult result = new StreamResult(new File("C:\\1\\file.xml"));
transformer.transform(source, result);

text.append(" Элемент создан!"+"\n");

} catch (ParserConfigurationException pce) {
pce.printStackTrace();
} catch (TransformerException tfe) {
tfe.printStackTrace();
```

ЛИСТИНГ ПЕРЕМЕЩЕНИЯ ЭЛЕМЕНТОВ

```
private void moved(int x, int y){//отрисовка фигур при переносе
if (currentShapeIndex >= 0) {
if(tableClick==-1)
{
temp = shapes[currentShapeIndex].getBounds();
checkshape = ShapeProperties[currentShapeIndex];
}
else{
temp = shapes[tableClick].getBounds();
checkshape = ShapeProperties[tableClick];
}
if(checkshape == 1)//квадрат
{
      temp.x = x-(temp.height/2);
      temp.y = y-(temp.height/2);
}
if(checkshape == 2)//прямоугольник
{
      temp.x = x-(temp.width/2);
      temp.y = y-(temp.height/2);
}
if(checkshape == 3)//круг
{
      tempx = x-(temp.height/2);
      tempy = y-(temp.height/2);
      temph = temp.height;
      tempw = temp.height;
}
if(checkshape == 4)//овал
{
      tempx = x-(temp.width/2);
      tempy = y-(temp.height/2);
      temph = temp.height;
      tempw = temp.width;
}
```

```
if(checkshape == 5 || checkshape == 9)//линия
{
      temp3 = (Line2D.Float) shapes[currentShapeIndex];
      tempx = x;//X1
      tempy = y;//Y1
      if ((int) temp3.getX1() < x)//X2
{
temph = (int) temp3.getX2()+(x-(int) temp3.getX1());
}
else
{
      temph = (int) temp3.getX2()-((int) temp3.getX1()-x);
}
if ((int) temp3.getY1() < y)//Y2
{
      tempw = (int) temp3.getY2()+(y-(int) temp3.getY1());
}
else
{
      tempw = (int) temp3.getY2()-((int) temp3.getY1()-y);
}
}
if(checkshape == 8)//треугольник
{
      temp2 = (Polygon) shapes[currentShapeIndex];
int height = (temp2.ypoints[1]-temp2.ypoints[0])/2;
int width = (temp2.xpoints[1]-temp2.xpoints[2])/2;
for(int i = 0;i<3;i++)
{
      temp2.xpoints[0] = x;
      temp2.ypoints[0] = y- height;
      temp2.xpoints[1] = x+width;
      temp2.ypoints[1] = y+ height;
      temp2.xpoints[2] = x-width;
      temp2.ypoints[2] = y+ height;
}
}if(checkshape == 6){
curvetemp = (QuadCurve2D) shapes[currentShapeIndex];
```

```
tempx = x;//X1
tempy = y;//Y1
if ((int) curvetemp.getX1() < x)//X2
{
     tempx2 = (int) curvetemp.getX2()+(x-(int) curvetemp.getX1());
}
else
{
     tempx2 = (int) curvetemp.getX2()-((int) cur-vetemp.getX1()-x);
}
if ((int) curvetemp.getY1() < y)//Y2
{
     tempy2 = (int) curvetemp.getY2()+(y-(int) curvetemp.getY1());
}
else
{
     tempy2 = (int) curvetemp.getY2()-((int) cur-vetemp.getY1()-y);
}
if ((int) curvetemp.getX1() < x)//X2
{
     ctrlx = (int) curvetemp.getCtrlX()+(x-(int) curvetemp.getX1());
}
else
{
     ctrlx = (int) curvetemp.getCtrlX()-((int) curvetemp.getX1()-x);
}
if ((int) curvetemp.getY1() < y)//Y2
{
     ctrly = (int) curvetemp.getCtrlY()+(y-(int) curvetemp.getY1());
}
else
{
ctrly = (int) curvetemp.getCtrlY()-((int) curvetemp.getY1()-y);
}
curvetemp = new
QuadCurve2D.Double(tempx,tempy,ctrlx,ctrly,tempx2,tempy2);
}
if(checkshape == 7){
```

```
qcurvetemp = (CubicCurve2D) shapes[currentShapeIndex];
tempx = x;//X1
tempy = y;//Y1
if ((int) qcurvetemp.getX1() < x)//X2
{
     tempx2 = (int) qcurvetemp.getX2()+(x-(int) qcurvetemp.getX1());
}
else
{
     tempx2 = (int) qcurvetemp.getX2()-((int) qcurvetemp.getX1()-x);
}
if ((int) qcurvetemp.getY1() < y)//Y2
{
     tempy2 = (int) qcurvetemp.getY2()+(y-(int) qcurvetemp.getY1());
}
else
{
     tempy2 = (int) qcurvetemp.getY2()-((int) qcurvetemp.getY1()-y);
}
if ((int) qcurvetemp.getX1() < x)//CTRLX1
{
     ctrlx = (int) qcurvetemp.getCtrlX1()+(x-(int) qcurvetemp.getX1());
}
else
{
     ctrlx = (int) qcurvetemp.getCtrlX1()-((int) qcurvetemp.getX1()-x);
}
if ((int)  qcurvetemp.getY1() < y)//CTRLY1
{
     ctrly = (int) qcurvetemp.getCtrlY1()+(y-(int) qcurvetemp.getY1());
}
else
{
     ctrly = (int) qcurvetemp.getCtrlY1()-((int) qcurvetemp.getY1()-y);
}
if ((int) qcurvetemp.getX1() < x)//CTRLX2
{
     ctrlx = (int) qcurvetemp.getCtrlX2()+(x-(int) qcurvetemp.getX1());
```

```
}
else
{
      ctrlx = (int) qcurvetemp.getCtrlX2()-((int) qcurvetemp.getX1()-x);
}
if ((int)  qcurvetemp.getY1() < y)//CTRLY2
{
ctrly = (int) qcurvetemp.getCtrlY2()+(y-(int) qcurvetemp.getY1());
}
else
{
      ctrly = (int) qcurvetemp.getCtrlY2()-((int) qcurvetemp.getY1()-y);
}
qcurvetemp = new CubicCurve2D.Double
(tempx,tempy,ctrlx,ctrly,ctrlx2,ctrly2,tempx2,tempy2);
}
repaint();
```

ПРИЛОЖЕНИЕ В

Принцип работы алгоритмов приложения

Процесс записи начинается с выбора элемента на панели выбора. При этом переменной int drawmethod присваивается число, обозначающее конкретную переменную. Для создания элемента, необходимо зажать левую кнопку мыши (при этом записывается начальная точка(startX, startY), а также переменной int drawmethod присваивается значение 1(начало записи)), и путём перемещения мыши задать необходимый размер. Перетаскивая курсор (mouse dragged) будущий примитив динамически отрисовывается на панели пунктирной линией используя библиотеку `java.Draw`.

```
if (drawmethod == 1){
     if (drawstate == 1)//если мышь отпущена
     {
     ...
     }
}
g.drawRect(x, y, width, width);//отрисовка элемента при со-
здании
```

Во время динамической отрисовки переменные endX, endY перезаписываются. При отпуске кнопки мыши переменной `int drawmethod` присваивается значение 0, что означает начало записи в массив. Двигать курсор возможно в любую сторону, при этом начальные и конечные точки пересчитываются.

```
x = (startX < endX) ? startX : endX; // если начальная точка по X
меньше конечной, то переменной присваивается переменная startX, d
противном случае endX.
y = (startY < endY) ? startY : endY; // если начальная точка по Y
меньше конечной, то переменной присваивается переменная startY, d
противном случае endY.
```

```
int width = endX - startX + 1; // переменной width присваивается разница конечной и начальной точки. В случае отрицательного результата, меняем знак.
if (width < 0) width = -width;
int height = endY - startY + 1; // переменной height присваивается разница конечной и начальной точки. В случае отрицательного результата, меняем знак.
if (height < 0) height = -height;
```

Исходя из метода записи (значения переменной drawmethod), выбирается класс для записи в массив shapes, и в массив записывается конкретный shape:

```
if (drawmethod == 1)
{
     ...
          shapes[numOfShapes] = new Rectangle(x, y, width, width);
     ...
}
```

Переменная numOfShapes увеличивается на единицу. На этом процесс записи в массив завершён.

Процесс записи свойств

После записи примитива в массив и записи информации, о его создании происходит запись свойств конкретного элемента. Для этого при записи элемента выполняются 2 функции:

```
addProperties(g);
addLayer(0, numOfShapes);//передаётся значение «элемент – rectangle, количество примитивов»
```

Функция `addProperties(g)` предполагает запись в массив `ShapeProperties[numOfShapes]` значения переменной drawmethod, что в дальнейшем будет необходимо для редактирования примитива. В переменную currentShapeIndex ответственную за обозначение текущего активного примитива записывается значение переменной numOfShapes для обозначения последнего созданного примитива

Функция `addLayer();` добавляет в табличную модель `DefaultTableModel dtm` запись, которая состоит из 2 столбцов и содержит:

```
public void addLayer(int z, int v){
     dtm.addRow(new Object[] {typeOfShape[z], v-1});
}
```

Массив String[] typeOfShape содержит строчные наименования примитивов, что необходимо для записи всех данных в файл xml:

```
String[] typeOfShape = {"rect", "ellipse", "line"...}
```

Число из второго столбца необходимо для указания порядкового номера элемента, а также для его поиска при нажатии на строку таблицы.

```
Public void mouseClicked(MouseEvent e) {
jTabProperties.clearSelection();//сброс предыдущих выделений
int row = jTabProperties.rowAtPoint(e.getPoint());
int col = jTabProperties.columnAtPoint(e.getPoint());
...
if (row >= 0 && col >= 0) {
tableClick = (int) jTabProperties.getValueAt(row, 1);
...
temp = shapes[tableClick].getBounds();
...
select = 1;
System.out.println(tableClick);
     }
}
```

Процесс поворота примитива

Процесс поворота примитива начинается с нажатия правой кнопки мыши на выделенном элементе. При этом номер элемента фиксируется в переменной int currentShapeIndex. Затем в методе void rotateShape() производятся манипуляции над элементом, в соответствии с его идентификатором.

Примитивы, обладающие одной опорной точкой и переменными, содержащими длины, такие как ellipse и rectangle поворачиваются на 90° путём перемены местами переменных height и width.

```
temp = shapes[currentShapeIndex].getBounds();//в переменную Rec-
tangle temp копируются координаты прямоугольника
int x = 0;// переменная для  временного хранения координаты
x = temp.height;//перемена местами длины и ширины
temp.height = temp.width;
temp.width= x;
shapes[currentShapeIndex] = temp;//запись новых координат в массив
```

Для примитивов, обладающих несколькими опорными точками, таких как линия или полигон требуется трансляция координат, как это показано на рисунке.

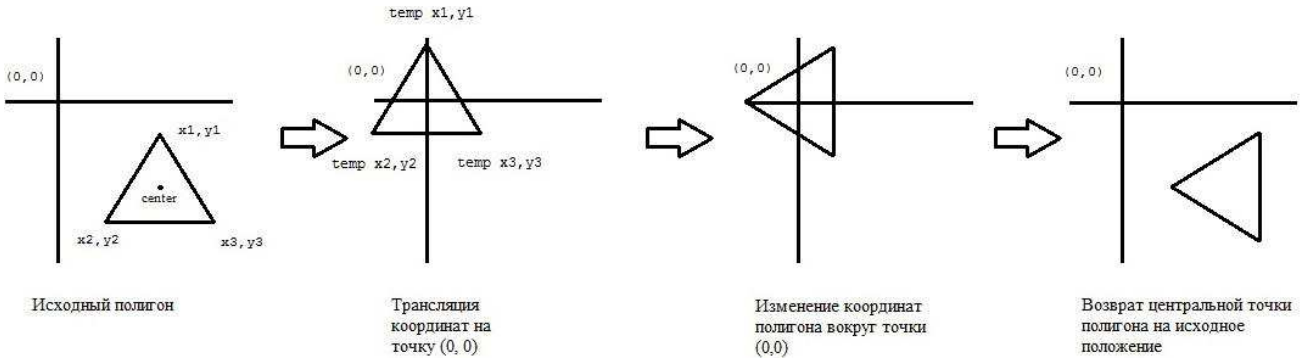


Рисунок 1 – процесс поворота примитива

Код для поворота координат выглядит следующим образом:

```
//Трансляция на начало координат
x1 = g.getX() - center.x;
y1 = g.getY() - center.y;
//Поворот
temp_x1 = x1 * Math.cos(angle) - y1 * Math.sin(angle));
temp_y1 = x1 * Math.sin(angle) + y1 * Math.cos(angle));
//Возврат координат на прежнее место
g.setLocation(temp_x1 + center.x, temp_y1 + center.y);
```

В данной работе полигон выступает в качестве треугольника. Точки X и Y хранятся в массивах `xpoints[] и ypoints[]`

```
temp2.xpoints[i] = (int) (centerx + (temp2.xpoints[i]-
centerx)*Math.cos(5) - (temp2.ypoints[i]-centery)*Math.sin(5));
temp2.ypoints[i] = (int) (centery + (temp2.xpoints[i]-
centerx)*Math.sin(5) + (temp2.ypoints[i]-centery)*Math.cos(5));
```

Поворот линии осуществляется подобным способом.

Процесс поворота примитива

Сохранение файла происходит посредством DOM парсера. Для парсинга используется библиотека `org.w3c.dom`. Каждый примитив записывается внутри тега `<gp:/>` в соответствии с его наименованием, содержащимся в массиве `string [] typeOfShape`. Идентификация примитива происходит с помощью массива `int [] ShapeProperties`, в котором содержится идентификатор типа примитива.

```
Element part = doc.createElement("gp:"+ typeOf-
Shape[ShapeProperties[i]-1]);
if(typeOfShape[ShapeProperties[i]-1] == "rect")
{
     temp = shapes[i].getBounds();
     part.setAttribute("x", Integer.toString(temp.x));
     part.setAttribute("y", Integer.toString(temp.y));
     part.setAttribute("width", Integer.toString(temp.width));
     part.setAttribute("height", Inte-
ger.toString(temp.height));
}else if(typeOfShape[ShapeProperties[i]-1] == "ellipse"){
     ...
}
```

Весь листинг парсера содержится в приложении Б.

ПРИЛОЖЕНИЕ Г

Правила описания графических элементов

Созданный элемент представляет собой XML-документ, в котором описаны примитивы, их свойства, и тип связи с элементами (пины и их типы). Также внутри документа описана идентификационная информация, такая как тип элемента, его класс и его ID. Данный XML-документ будет использован приложении ECEditorFX, где описаны правила их структуры в документах .xsd[18].

Пример элемента, полученного по результату работы программы:

```
<el:element  id="r555" name="resistor"…>
     <el:subClass subClassID="7" subClassName="RESISTORS"/>
     <el:type typeID="2" typeName="BASE_RESISTOR"/>
     <el:center x="240.0" y="180.0"/>
     <el:view>
          <gp:g>
               <gp:rect height="116" width="201" x="141" y="121"/>
               <gp:line x1="240.0" x2="240.0" y1="40.0" y2="320.0"/>
          </gp:g>
     </el:view>
     <el:pins>
          <el:pin id="i1.p0" type="IN" visible="true">
               <ut:center x="240" y="41"/>
          </el:pin>
          <el:pin id="i1.p0" type="OUT" visible="true">
               <ut:center x="240" y="320"/>
          </el:pin>
     </el:pins>
</el:element>
```

Приведенный выше пример соответствует графическому отображению элемента, представленному на Рисунок .

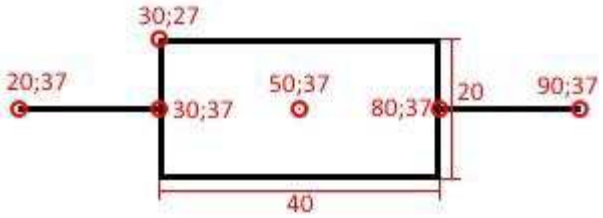


Рисунок 1 — Графическое отображение элемента

На рисунке 1 обозначены точки каждого примитива из которого строится графическое отображение объекта. Точка 50;37 соответствует середине элемента и указана в примере в атрибутах элемента "center".

Описание элементов:

– element – корневой XML-элемент, внутри которого содержатся XML-элементы описывающие объект. Содержит атрибут "id" – обозначающий идентификатор объекта, принимающий строковое значение и имеющий тип "ID", и атрибут "name" – обозначающий название описываемого элемента, и принимающий значения типа "string";

– subClass – информация о принадлежности элемента подклассу, в атрибутах "subClassName" и "subClassID" указывается название и идентификатор подкласса элемента, атрибуты принимают значения типа "string";

– type – тип элемента, в атрибутах "typeName" и "typeID" указывается название и идентификатор типа элемента;

– center – описывает координаты центра элемента, атрибуты "x" и "y" принимают значения типа "float";

– view – корневой XML-элемент для XML-элементов описания графического отображения объекта;

– pins – корневой XML-элемент для XML-элементов "pin" описывающих точки подключения элемента;

– param – элемент содержит текстовое значение параметра элемента, принимает значения типа "string";

– text – текстовое содержание элемента, принимает значение типа "string".

Федеральное государственное автономное
образовательное учреждение
высшего образования
"СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ"

Институт космических и информационных технологий

Вычислительная техника

УТВЕРЖДАЮ
Заведующий кафедрой ВТ
_____А.И.Легалов
"20" июня 2017г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Подсистема формирования графических элементов для системы активного тестирования

09.04.01 Информатика и вычислительная техника
09.04.01.02 Информационное и программное обеспечение САПР

Научный руководитель	_____ подпись, дата	проф., д-р тех. наук *должность, ученая степень*	С.А.Бронов
Выпускник	_____ подпись, дата		Д.М.Севостьянов
Рецензент	_____ подпись, дата	проф., д-р тех. наук *должность, ученая степень*	Ю.В.Краснобаев
Нормоконтролер	17.06.17 подпись, дата	доц., канд. тех. наук *должность, ученая степень*	В.И.Иванов

Красноярск 2017