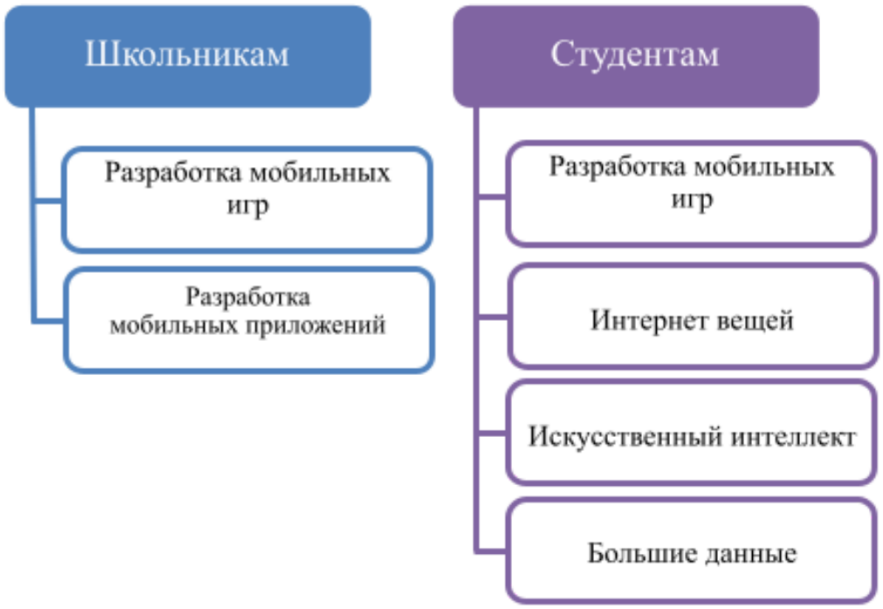
Образовательные программы Samsung

Кампус инноваций Samsung – глобальный социально-образовательный проект компании Samsung Electronics (далее **Проект**), направленный на подготовку молодых специалистов по востребованным направлениям ИТ-индустрии.

Образовательные треки Проекта:



Партнеры Проекта (далее Партнеры) – образовательные организации высшего, общего, среднего специального или дополнительного образования, связанные с компанией Samsung Electronics договорными отношениями по совместной реализации Проекта в формате включения в учебную программу Партнера. Полный перечень Партнеров публикуется на учебном портале Проекта.

Программа обучения по курсу трека определяет требования ко входному уровню учащихся, содержание, рекомендуемый объем часов, формат проведения занятий при реализации учебного процесса Партнером Проекта. Обучение учащихся по Программе бесплатное.

Учебный портал с входом в систему электронного обучения: https://innovationcampus.ru/lms/.

Электронная почта: info@innovationcampus.ru.

Программа обучения «Разработка мобильных игр»

Курс "Разработка мобильных игр" разработан для школьников 8-9 классов, стремящихся освоить основы программирования и интересующихся индустрией видеоигр. Одной из ключевых особенностей курса является его практико-ориентированность: изучение одного из самых востребованных на рынке языков программирования Java построено на создании игр для смартфонов на платформе Android.

Технологии, рассматриваемые в курсе:

- Язык программирования Java,
- Интегрированные среды разработки IntelliJ IDEA и Android Studio,
- Фреймворк LibGDX,
- Библиотеки Box2D, Scene2D и FreeType.

Особенностями программы являются:

- **Практическая направленность**. Получение практического результата на каждом шаге траектории обучения с приведением релевантных теоретических материалов.
- **Широкая аудитория**. Доступное и детальное изложение материала позволяет снизить требования к уровню начальной подготовки учащихся для успешного освоения курса. С другой стороны, в курсе встречаются уточняющие вопросы и дополнительные материалы, которые предназначены для учащихся, готовых к более продвинутому уровню обучения.
- **Формат сторителлинга** делает обучение более увлекательным, погружая ученика в ситуацию прохождения стажировки в гейм-студии. Этот подход способствует удержанию внимания учащихся и облегчает восприятие сложных концепций через интересные и понятные объяснения.
- **Проектное обучение**: разработка учащимся игровых проектов в рамках спринтов*, а также собственной игры в виде командного проекта.
- **Профориентационный характер.** Программа курса охватывает все основные аспекты разработки игр, включая программирование, работу с графикой и физикой, создание пользовательского интерфейса. Такой подход обеспечивает всестороннюю подготовку учащегося к реальным задачам в области разработки игр.

Продолжительность обучения 150 ак. час., из них: 48 — аудиторные занятия по 4 часа в неделю, 8 часов — консультации по выполнению проектов. 94 часа отводится на самостоятельное прохождение программы в учебной системе.

Обучение поделено на 3 спринта[1] по 4 недели каждый (по 4 академических часа очных занятий в неделю). После завершения всех спринтов на реализацию выпускного проекта выделяется 2 недели.

Ежегодные сроки старта и продолжительности курса: сентябрь – декабрь, февраль – май.

Требования к начальной подготовке учащихся

Программа ориентирована на школьника 8-9 класса, который:
− имеет склонность к алгоритмическому мышлению, увлекается ИТ-технологиями;
− имеет интерес к индустрии видеоигр.

Цель освоения курса

Обучить основам программирования на примере языка Java через создание мобильных игр с использованием фреймворка LibGDX.

Планируемые результаты освоения курса

После успешного завершения курса выпускник сможет:

Программировать на Java:

- Применять основные концепции объектно-ориентированного программирования (ООП) в контексте разработки игр.
- Работать с классами, объектами, наследованием, полиморфизмом и интерфейсами

Разрабатывать мобильные игры:

- Использовать фреймворк LibGdx для создания 2D игр.
- Реализовывать основные механики игр, включая обработку ввода пользователя, управление персонажами, работу с анимацией и физикой.

Работать с графикой и анимацией:

- Создавать и интегрировать графические ресурсы в игровой проект.
- Управлять текстурами, шрифтами и другими графическими элементами.

1 *В рамках программы учащиеся разрабатывают игровые проекты на каждом этапе обучения, который мы называем "спринтом". Этот термин заимствован из методологии Scrum, популярной в ИТ-индустрии, где он обозначает короткие, концентрированные периоды работы над конкретными задачами. Мы внедряем элементы Agile и Scrum в образовательный процесс, чтобы познакомить учащихся с проектной деятельностью в формате, приближенном к реальной практике разработки программного обеспечения. В результате, на каждом "спринте" ученики создают отдельные игровые проекты.*

Управлять игровыми процессами:

- Разрабатывать игровые циклы, состояния игры и систему уровней.
- Реализовывать взаимодействие игровых объектов и обработку событий.

Создавать пользовательские интерфейсы:

- Программировать интерфейсы для мобильных игр, включая меню, кнопки, и другие элементы взаимодействия.

Анализировать процессы разработки игр:

- Анализировать процесс создания игр, выявляя основные этапы и необходимые ресурсы.
- Анализировать ошибки и проблемы, возникающие в ходе разработки игр, и находить эффективные способы их решения.

Описывать игровые приложения:

- Описывать функциональность и особенности разработанных игровых приложений.

Краткое содержание курса

- **Спринт 1. Введение в Java**
  В рамках данного спринта учащиеся познакомятся с основами программирования на языке Java. Будут рассмотрены ключевые концепции, такие как платформы и кроссплатформенность, работа с примитивными типами данных, условные конструкции, циклы, массивы, а также основы объектно-ориентированного программирования (ООП). Практическая часть включает создание простого игрового проекта в виде консольного квеста, где учащиеся смогут применить изученные концепции на практике: от создания и управления игровым полем до разработки базовых игровых механик, таких как управление персонажами и обработка событий.

- **Спринт 2. Знакомство с LibGDX**
  Во втором спринте учащиеся погружаются в разработку мобильных игр с использованием фреймворка LibGDX. В процессе обучения участники изучат базовые компоненты фреймворка, такие как жизненный цикл игры, классы для работы с графикой и анимацией, а также научатся реализовывать адаптивный интерфейс для различных устройств. Практическая часть включает создание простых игровых механик, таких как управление персонажем, обработка столкновений, анимация, начисление очков, работа с пользовательским интерфейсом и реализация параллакса. По завершении спринта учащиеся разработают собственную версию игры-кликера, вдохновленную популярной игрой "Flappy Bird".

- **Спринт 3. Работаем с Box2D**
  В третьем спринте участники познакомятся с физическим движком Box2D и научатся интегрировать его в игровое приложение. В ходе занятий они будут работать с коллайдерами, телами, миром и силами, а также разберутся с обработкой столкновений. Участники реализуют интерфейс игры, включая добавление текстур и пользовательских элементов, изучат методы работы с шрифтами, музыкальным сопровождением и звуками. На практике учащиеся создадут игровой проект с использованием физических взаимодействий, а также разработают систему пауз, рекордов и сохранений. Прототипом игры для третьего спринта является “Space invaders”.

- **Спринт 4. Создаем платформер (Дополнительный)**
  Четвертый спринт посвящен разработке платформера. Участники изучат функционал библиотеки Scene2d для работы с интерфейсом и сценами, а также освоят создание уровней игрового мира с помощью тайлов и инструментов для построения карт. В ходе обучения они будут работать с Viewport, управляя адаптацией изображения под различные экраны, и внедрят

новые механики с использованием Box2D. Также будет изучено применение паттернов проектирования для оптимизации кода и улучшения производительности игры. Этот спринт не входит в основную программу и предназначен для самостоятельного изучения обучающимися.

Элементы контроля и система оценивания

Задания для самоконтроля

Выполнение практических заданий в ходе занятий, связанных с изучением и применением различных аспектов разработки игр. Баллы за тестирования **не** влияют на итоговый результат по программе, они созданы с целью выявления проблемных областей в понимании материала.

Промежуточный контроль – игры в спринтах

Для успешного прохождения каждого спринта обучающемуся необходимо реализовать игру в соответствии с предоставленным в курсе техническим заданием. Реализация игры обучающимся должна демонстрировать освоение знаний и навыков, изучаемых в данном спринте.
Максимальная оценка за проект после спринта составляет 10 баллов: 8 баллов за реализованную игру и 2 балла за активность в спринте. Проект оценивается вручную преподавателем. Спринт считается успешно пройденным, если обучающийся набрал не менее 5 баллов, при этом оценка преподавателя за проект должна составлять не менее 1 балла. При необходимости, проект может быть сдан повторно.

Итоговый контроль – финальный проект

Финальный проект, включающий разработку и представление собственной игры, демонстрирующей освоение всех изученных концепций и технологий курса. Выполняется индивидуально или в команде из 2-3 человек. Тема определяется учащимися.

Формат выполнения и оценивания финального проекта выбирается администрацией площадки. Предусмотрены два варианта.

1. Общий формат
Проект оценивается приглашёнными экспертами в ходе защиты. Учитель группы также может выступать в качестве одного из экспертов.

Максимальная оценка за проект составляет 10 баллов и рассчитывается как среднее арифметическое оценок, выставленных всеми экспертами, присутствующими на защите.

Финальный проект считается успешно выполненным, если обучающийся набрал не менее 5 баллов.

2. Предпочтительный формат с кросс-проверкой

В оценивании проекта участвуют приглашённые эксперты, учащиеся других команд и члены команды, выполнившей проект.

Максимальная оценка за проект составляет 10 баллов и рассчитывается по следующей формуле:

0,5 * X + 0,3 * Y + 0,2 * Z, где

- X (от 0 до 10) – среднее арифметическое оценок, выставленных всеми экспертами, присутствующими на защите. Учитель группы также может выступать в качестве одного из экспертов;
- Y (от 0 до 10) – оценка проекта учащимися других команд в рамках кросс-проверки;
- Z (от 0 до 10) – оценка вклада обучающегося в работу команды, выставленная другими членами его команды по установленным критериям. Значение Z рассчитывается как среднее арифметическое оценок, полученных от других членов команды.

Финальный проект считается успешно выполненным, если обучающийся набрал не менее 5 баллов.

Общий балл по программе рассчитывается по следующей формуле:

0,5 * (A + B), где

- A (от 0 до 10) – оценка за финальный проект.
- B (от 0 до 10) – оценка за прохождение спринтов. Рассчитывается как среднее арифметическое всех полученных баллов за спринты.

Сертификат о прохождении программы выдается при общем балле не менее 5. Сертификат с отличием – не менее 9.

Обязательным условием получения сертификата является успешное прохождение всех спринтов и финального проекта. Если хотя бы один спринт или финальный проект не пройден, сертификат не выдается независимо от общего балла.

Возможности для выпускников курса

После успешного завершения программы перед учащимися открываются следующие возможности:

1. Участие в олимпиадах и конкурсах по направлениям, связанным с разработкой игр.
2. Разработка собственных игровых проектов.
3. Продолжение обучения на продвинутых курсах и специализированных программах в области разработки игр, мобильной разработки и программирования.
4. Участие в хакатонах и геймджемах.
5. Создание портфолио игровых проектов, что может стать важным шагом для начала карьеры в индустрии игр.
6. Несмотря на прикладную направленность с целью профориентации, изучение курса дает возможность заложить фундаментальные знания и навыки в программировании, что позволит в дальнейшем осваивать новые технологии и инструменты в ИТ-направлениях, не связанных с разработкой и дизайном игр.

Тематический план курса

№	Часов самостоятельной работы	Часы очных занятий	Описание темы	Практикум
1	20	12	**Спринт 1. Введение в Java**	
1.1	2	2	**Встреча 1. Кто-то сказал программирование?** Понятия платформы и кроссплатформенности, что из себя представляет язык программирования Java. JVM, JDK. Начало работы с интегрированной средой разработки IntelliJ IDEA.	Создание проекта в IntelliJ IDEA. Запуск проекта.
1.2	2		**Встреча 2. «Кажется, я не сохранил название в переменную…»** Переменные, примитивные типы данных, инициализация, присваивание, объявление. Правила работы с переменными. Арифметические операции. Строки. Ввод/вывод данных.	Инициализация основных характеристик игры. Создание игрового поля. Вывод игровой информации в консоль. Получение данных от игрока из консоли.
1.3	2	2	**Встреча 3. Железные логические операции** Условные конструкции: if-else, switch. Логические операции: конъюнкция, дизъюнкция, инверсия.	Обработка данных от пользователя. Работа с игровым меню. Отрисовка персонажей на поле. Логика передвижения персонажа. Проверка хода.
1.4	2	2	**Встреча 4. Массивы и циклы или «как сделать многомногомного…»** Понятие цикла. Итеративные конструкции while, do-while, for. Массивы: одномерные и двумерные. Вложенные циклы. Оператор перехода break.	Длительность игровой сессии. Обновление игрового поля. Отображение передвижения персонажа.
1.5	2	2	**Встреча 5. Коротко и понятно: методы работы с методами** Понятие метода. Виды методов: возвращающие и не возвращающие значения. Модификатор static. Понятия аргумента и параметра. Передача параметров, возвращение результата. Видимость переменных.	Обработка столкновения игрока и монстра. Создание квестов.
1.6	3	2	**Встреча 6. ООП – отличный описательный прием** Понятие и необходимость объектно-ориентированного программирования. Класс, объект, как экземпляр класса, поле, метод, конструктор. Перегрузка методов. Доступ к полям и методам класса. Модификаторы доступа.	Выделение главного героя в отдельный класс.
1.7	3	2	**Встреча 7. Принципы ООП или знакомство с китами** Инкапсуляция, наследование, полиморфизм. Переопределение метода.	Выделение монстра в отдельный класс. Добавление нового вида монстров.
1.8	*4*		*Доработка проекта* *	

№	Часов самостоя-тельной работы	Часы очных занятий	Описание темы	Практикум
2	22	12	**Спринт 2. Знакомство с LibGDX**	
2.1	3	2	**Встреча 1. LibGDX, заводи мотор** Понятие фреймворка. Начало работы с Android Studio. SDK. Структура проекта LibGDX. Запуск проекта на смартфоне, десктопе и эмуляторе. Класс SpriteBatch. Жизненный цикл игры. Класс Texture.	Генерация проекта. Запуск проекта. Изменение изображение. Работа с координатами птички.
2.2	3	2	**Встреча 2. На какой экран навести камеру?** Понятие адаптивности. Реализация констант в Java. Экраны. Интерфейс. Описание методов интерфейса Screen. Класс OrthographicCamera. Жизненный цикл экрана. Модификаторы final и static. Класс Game.	Реализация адаптивности. Разделение общей игровой логики на несколько экранов.
2.3	2	2	**Встреча 3. Учим птичку прыгать** Создание анимаций. Обработка нажатия на экран.	Вынос логики птички в отдельный класс. Обработка нажатия на экран. Анимация птички.
2.4	2	2	**Встреча 4. Преодолевая препятствия** Понятие пользовательского интерфейса (UI). Переиспользование объектов.	Реализация класса колонны.
2.5	2	2	**Встреча 5. Учимся проигрывать** Понятие коллизии. Обработка коллизии.	Реализация столкновения.
2.6	2		**Встреча 6. Считаем игровые очки** BitmapFont. Порядок отрисовки элементов.	Реализация логики начисления игровых очков. Отображение игровых очков на экране.
2.7	1	2	**Встреча 7. Игровые пейзажи** Понятие параллакса. Реализация параллакса в LibGDX.	Добавление фона. Реализация параллакс-эффекта.
2.8	2		**Встреча 8. Попробуем всё начать сначала?** Приведение: явное и неявное. Класс GlyphLayout. Метод unproject().	Добавление экранов для меню и рестарта. Добавление элементов пользовательского интерфейса.
2.9	1		***Встреча 9. Последний рывок***	*Создание игрового меню в формате самостоятельной работы.*
2.10	*4*		*Доработка проекта* *	

№	Часов самостоя-тельной работы	Часы очных занятий	Описание темы	Практикум
3	28	12	**Спринт 3. Работаем с Box2D**	
3.1	3	2	**Встреча 1. Первые мгновения после большого взрыва** Понятие библиотеки. Необходимость и функционал библиотек FreeType и Box2D. Класс ScreenAdapter.	Генерация проекта. Реализация адаптивности.
3.2	3		**Встреча 2. Вскрываем ящик Пандоры** Диаграммы UML. Классы World, Body и Fixture. Понятие коллайдера. Методы работы с миром и телами в box2d.	Реализация класса корабля. Инициализация базовых характеристик миру и телам.
3.3	3	2	**Встреча 3. Засоряем космос** Понятие игровой сессии. Работа со временем. Класс TimeUtils. Работа с динамическими массивами в Java (ArrayList). Понятие класса-оболочки.	Реализация игровой сессии. Реализация класса мусора.
3.4	1	2	**Встреча 4. Космическое оружие** Работа с физическим телом “пуля” в box2d.	Реализация класса лазера.
3.5	2		**Встреча 5. Есть контакт!** Слушатель контактов в box2d. Понятие колбека. Класс Contact. Идентификация физических тел при столкновении. Методы класса Fixture.	Обработка коллизий.
3.6	2	2	**Встреча 6. Космический интерфейс** Собственные шрифты в LibGDX (библиотека FreeType). Разделение элементов UI. Ключевое слово null в Java.	Реализация пользовательского интерфейса. Добавления фона в игру.
3.7	2		**Встреча 7. Время поставить все на паузу** Перечисления (enum) в Java. Состояния игровой сессии.	Реализация паузы. Обработка нажатий на кнопки.
3.8	1	2	**Встреча 8. Игровое меню** Работа с UI элементами. Смена игровых состояний.	Реализация игрового меню. Работа с состояниями сессии.
3.9	2		**Встреча 9. Первые сигналы из космоса** Работа со звуком и музыкой. Классы Music и Sound.	Добавление музыкального и звукового сопровождения в игру. Реализация класса настроек.
3.10	3	*2*	**Встреча 10. Лучшие из лучших** Запись и чтение данных при помощи класса Preferences.	Реализация таблицы рекордов. Сохранение результатов и настроек.
3.11	*6*		*Доработка проекта* *	

№	Часов самостоятельной работы	Описание темы	Практикум
4	30	**Спринт 4. Создаем платформер (Дополнительный)**	
4.1	3	**Время выйти на сцену** Инструментарий Scene2D. Классы Stage и Actor. Viewport и адаптивность интерфейса. Skin и UI-компоненты Table, Label, TextButton, List, ScrollPane и Image. Обработка нажатий и создание покадровой анимации.	Реализация базовой архитектуры экранов, главного меню, экрана настроек и анимированного фона.
4.2	1	**Учимся правильно работать со строками** Проблема строк, жёстко заданных в коде. Понятия интернационализации и локализации. Класс I18NBundle. Файлы формата properties. Работа с ключами и динамическими параметрами строк.	Создание файла строковых ресурсов. Перенос текстов интерфейса из Java-кода и их загрузка с помощью I18NBundle.
4.3	2	**Укладываем плитку** Карты и тайлсеты Tiled. Тайловые слои и слои объектов. Классы TmxMapLoader, TiledMap и OrthoCachedTiledMapRenderer. Хранение прогресса с помощью Preferences.	Реализация системы уровней, сохранение их состояния, формирование списка доступных уровней, загрузка и отображение выбранной карты.
4.3*	2	**Разрабатываем уровень в Tiled Map Editor (Дополнительная)** Интерфейс и основные инструменты Tiled Map Editor. Понятия тайла, тайлсета, карты, слоя и объекта. Форматы TMX и TSX. Пользовательские свойства объектов.	Создание собственного уровня по GDD: настройка карты и тайлсета, размещение платформ, коллайдеров, персонажей и интерактивных объектов.
4.4	3	**Строим мир** Паттерн проектирования Builder и цепочки вызовов. Классы Body, Fixture, CircleShape и PolygonShape. Связь актёров Scene2D с физическими телами Box2D. Разделение игровых объектов на персонажей и элементы окружения.	Реализация билдера физических объектов и базовой иерархии классов для игрока, врагов и статических элементов уровня.
4.5	2	**Заселяем мир** Парсинг TMX-карт. Работа со слоями, объектами RectangleMapObject и пользовательскими свойствами. Перечисления для описания слоёв и объектов карты. Управление физическим миром и очистка ресурсов.	Реализация B2WorldManager. Автоматическое создание стен, пола, игрока и врагов по данным из Tiled, добавление актёров на сцену и подготовка мира к перезапуску уровней.
4.6	3	**Добавляем движение** Абстракция пользовательского ввода. Метод handleInput(). Определение платформы запуска приложения. Управление на компьютере и Android. Работа с силами и скоростью тел Box2D. Состояния и анимации персонажей.	Реализация бега, прыжка и атаки игрока, мобильного контроллера, движения камеры, патрулирования врагов и эффекта параллакса для игрового фона.
4.7	4	**Ловим события** Понятие игровой сессии и состояния PLAYING, PAUSED и ENDED. Событийный подход и слушатели. Интерфейсы Hittable и ContactListener. Категории коллизий, сенсоры и безопасное удаление тел из мира Box2D.	Реализация получения урона и поражения персонажа, уничтожения врагов, начисления очков, проигрыша при падении в яму и победы при достижении финиша. Добавление ямы и финишного флага,

			сохранение прохождения уровня.
4.8	4	**Последние штрихи** Обработка завершения контактов. Дополнительные фикстуры головы и ног персонажа. Взаимодействие с поверхностями и лестницами. Компоненты HUD и диалоговые окна Scene2D.	Реализация алмазов, лестниц и бонусных блоков. Отображение очков, оставшихся жизней и времени прохождения. Добавление окон паузы, победы и поражения, а также логики перезапуска уровня, продолжения игры и возвращения в главное меню.
4.9	*6*	*Доработка проекта* *	

** Возможные варианты доработки игры каждого спринта предложены в учебной системе.*