Учреждение образования
«Гомельский государственный университет
имени Франциска Скорины»

Факультет физики и информационных технологий
Кафедра автоматизированных систем обработки информации

СОГЛАСОВАНО
Заведующий кафедрой
автоматизированных систем
обработки информации
А.В.Воруев
__ _____ 2023 г.

СОГЛАСОВАНО
Декан
факультета физики и
информационных технологий
Д.Л.Коваленко
__ _____ 2023 г.

ЭЛЕКТРОННЫЙ УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

ПРОГРАММИРОВАНИЕ ОБЛАЧНЫХ СЕРВИСОВ

для учащихся второй ступени высшего образования (магистратура)
специальности 1-45 80 01 Системы и сети инфокоммуникаций

составители: доцент кафедры АСОИ, к.ф.-м.н., Бычков П.В.
доцент кафедры ПОИТ БГУИР, к.т.н, доцент, Левчук В.Д.
ассистент кафедры АСОИ Рафалова Е.В.

Рассмотрено и утверждено
на заседании кафедры АСОИ
__14 марта___ 2023 г., протокол № _8_

Рассмотрено и утверждено
на заседании научно-методического
совета университета
30. 03. 2023 г., протокол № 7

Гомель 2023

1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Электронный учебно-методический комплекс (ЭУМК) по дисциплине «Программирование облачных сервисов» представляет собой комплекс систематизированных учебных, методических и вспомогательных материалов, предназначенных для использования в образовательном процессе специальности 1-45 80 01 Системы и сети инфокоммуникаций.

ЭУМК разработан в соответствии со следующими нормативными документами:

1. Положением об учебно-методическом комплексе на уровне высшего образования, утвержденном постановлением Министерства образования Республики Беларусь от 26.07.2011 №167.

2. Учебного плана УВО специальности высшего образования второй ступени (магистратура) 1-45 80 01 Системы и сети инфокоммуникаций регистрационный № I 45-2-01/Д-19 от 09.04.2019 г.

3. Учебной программой по учебной дисциплине «Протоколы дистанционного управления» для специальности 1-45 80 01 Системы и сети инфокоммуникаций, утвержденной 22.05.2019, регистрационный номер УД-31-2019-643/уч.

Целью дисциплины «Программирование облачных сервисов»является овладение основами программируемого доступа к облачным сервисам и ресурсам; изучение теоретических основ создания приложений для облачных сервисов;получение практических навыков создания приложений для облачных сервисов.

ЭУМК направлен на всестороннюю подготовку учащихся теоретическим основам и практическим навыками по решению новых инженерных задач, возникающих при использовании теоретических аспектов программирования web-приложений для их реализации; управления прикладным программным обеспечением для настройки и развертывания приложений в облачном сервисе.Организация изучения дисциплины на основе ЭУМК предполагает продуктивную образовательную деятельность, позволяющую сформировать социально-личностные и профессиональные компетенции будущих специалистов.

ЭУМК способствует успешному осуществлению учебной деятельности, дает возможность планировать и осуществлять самостоятельную управляемую работу учащихся, обеспечивает рациональное распределение учебного времени по темам учебной дисциплины и совершенствование методики проведения занятий.

ЭУМК состоит из теоретического, практического и вспомогательного разделов. Теоретический раздел содержит тексты лекций. Практический раздел содержит методические рекомендации к лабораторным работам, тестовые задания и вопросы для самоконтроля. Вспомогательный раздел содержит учебную программу и список литературы.

Теоретический раздел содержит лекционный материал по всем темам учебной программы, включая и темы, вынесенные на самостоятельное изучение. В разделе так же содержатся рекомендации по организации и выполнению управляемой самостоятельной работы по трем уровням сложности.

Практический раздел включает в себя темы лабораторных занятий и задания с краткими методическими указаниями по выполнению лабораторных работ. В разделе так же приводятся некоторый набор тестовых заданий и к каждой теми указаны вопросы для самоконтроля.

Вспомогательный раздел содержит необходимые элементы учебно-программной документации по дисциплине с указанием рекомендуемой литературы (основной, дополнительной, вспомогательной).

Все разделы ЭУМК в полной мере соответствуют содержанию учебной программы и объему учебного плана.

Дисциплина компонента УВО «Программирование облачных сервисов» изучается магистрантами 1 года обучения (1семестр) дневной формы обучения и 1 года обучения (1 семестр) заочной формы обучения для специальности: 1-45 80 01 Системы и сети инфокоммуникаций.

Общее количество часов – 230, зачетных единиц – 6.

Дневная форма обучения: аудиторное количество часов – 72; из них: лекционных занятий – 32, практических занятий – 24, лабораторных работ – 16, (управляемая самостоятельная работа – 14).

Форма отчётности – зачет.

Заочная форма обучения: аудиторное количество часов – 16; из них: лекционных занятий – 6, практических занятий – 4, лабораторных работ – 6.

Форма отчётности – зачет, зачетных единиц – 6.

2 ТЕКСТЫ ЛЕКЦИЙ

Тема 1. Основы протокола HTTP

HTTP - это протокол, позволяющий получать различные ресурсы, например HTML-документы. Протокол HTTP лежит в основе обмена данными в Интернете. HTTP является протоколом клиент-серверного взаимодействия, что означает инициирование запросов к серверу самим получателем, обычно веб-браузером (web-browser).

Стартовая (начальная) строка запроса для HTTP 1.1 составляется по следующей схеме:

- Метод URI HTTP/Версия

Например (такая стартовая строка может указывать на то, что запрашивается главная страница сайта): `GET / HTTP/1.1`

Метод (в англоязычной тематической литературе используется слово method, а также иногда слово verb- «глагол») представляет собой последовательность из любых символов, кроме управляющих и разделителей, и определяет операцию, которую нужно осуществить с указанным ресурсом. Спецификация HTTP 1.1 не ограничивает количество разных методов, которые могут быть использованы, однако в целях соответствия общим стандартам и сохранения совместимости с максимально широким спектром программного обеспечения как правило используются лишь некоторые, наиболее стандартные методы, смысл которых однозначно раскрыт в спецификации протокола.

URI (UniformResourceIdentifier, унифицированный идентификатор ресурса) — путь до конкретного ресурса (например, документа), над которым необходимо осуществить операцию (например, в случае использования метода GET подразумевается получение ресурса). Некоторые запросы могут не относиться к какому-либо ресурсу, в этом случае вместо URI в стартовую строку может быть добавлена звёздочка (астериск, символ «*»). Например, это может быть запрос, который относится к самому веб-серверу, а не какому-либо конкретному ресурсу. В этом случае стартовая строка может выглядеть так:

- OPTIONS * HTTP/1.1

Версия определяет, в соответствии с какой версией стандарта HTTP составлен запрос. Указывается как два числа, разделённых точкой (например 1.1).

Для того, чтобы обратиться к веб-странице по определённому адресу (в данном случае путь к ресурсу - это «/»), нам следует отправить следующий запрос:

- GET / HTTP/1.1

```
Host: asoi.gsu.by
```

Если вы хотите отправить запрос в точном соответствии со спецификацией, можете воспользоваться управляющими последовательностями \r и \n:

```
    echo -en "GET / HTTP/1.1\r\nHost: asoi.gsu.by \r\n\r\n" | ncat asoi.gsu.by
80
```

Клиент и сервер. Полная форма URL. Формат запроса клиента. Методы. Формат ответа сервера.

Тема 2. Понятие и обзор облачных сервисов

Есть две классификации облачных услуг. Начнём с самой простой. Облака бывают:

Частные. ИТ-инфраструктура предназначена для использования одной организацией. Оборудование может находиться как на территории самой компании, так и и на арендованной территории в дата-центре.

Публичные. IT-инфраструктура облака принадлежит провайдеру и предоставляется компании-клиенту в аренду. Провайдер выделяет необходимые заказчику виртуальные ресурсы. У клиентов нет физического доступа к оборудованию.

Гибридные. Комбинация частной и публичной инфраструктуры. Часть оборудования может находиться в собственности пользователя, а часть - в публичном сервисе.

Вторая классификация более точечная и опирается на вид услуги, который предлагается провайдером. На рынке облачных сервисов есть разные предложения, которые подойдут под разные задачи. Поговорим о каждом из них в отдельности. И начнём с самых простых и знакомых нам услуг.

У каждого вида облачной услуги есть свои преимущества и недостатки, но мы выделим основные общие пункты.

Из плюсов:

- ответственность за работоспособность оборудования лежит на провайдере. Заказчику не нужно думать о том, как обеспечить бесперебойное питание, подключение к интернету, охлаждение;
- взять готовую настроенную услугу проще, чем выстраивать собственную инфраструктуру;
- арендовать мощности может быть дешевле, чем тратиться на собственные серверы.

Среди минусов:

- полная зависимость от поставщика. Пользователь зависим от провайдера и качества обслуживания его серверов. Проблемы у провайдера незамедлительно коснутся пользователя;
- нужно подключение к интернету. Без интернета пользователь не сможет работать с сервисом.

Также стоит отметить, что облачные услуги не подходят для предприятий, имеющих отношение к государственной и военной тайне.

Определение облачного хранилища и сервиса. Виды облачных сервисов. GoogleDrive, MicrosoftOneDrive, Dropbox, Яндекс.Диск.

Тема 3. Обзор языка JS

Базовый язык JavaScript стандартизирован комитетом ECMA TC-39 как язык программирования под названием ECMAScript.Базовый язык также используется в не-браузерном окружении, например, в node.js.

Программные интерфейсы приложения (API), встроенные в браузеры, обеспечивающие различные функциональные возможности, такие как динамическое создание HTML и установку CSS стилей, захват и манипуляция видеопотоком, работа с веб-камерой пользователя или генерация 3D графики и аудио сэмплов.

Сторонние API позволяют разработчикам внедрять функциональность в свои сайты от других разработчиков, таких как Twitter или Facebook.

Также вы можете применить к вашему HTML сторонние фреймворки и библиотеки, что позволит вам ускорить создание сайтов и приложений.

Переменные. Операции. Логические операторы. Операторы цикла. Перехват ошибок.

Объявление. Локальные переменные. Внешние переменные. Параметры. Аргументы по умолчанию. Возврат значения. Функциональные выражения.

Тема 4. Данные в языкеJS

Объекты как ассоциативные массивы. Операции с объектом. Перебор свойств. Передача по ссылке. Копирование по значению и ссылке. Клонирование объектов.

Глобальный объект. Замыкания, функции изнутри. [[Scope]] для newFunction. Локальные переменные для объекта. Модули через замыкания. Управление памятью.

Все используемые данные в javascript имеют определенный тип. В JavaScript имеется восемь типов данных:

- String: представляет строку

- Number: представляет числовое значение
- BigInt: предназначен для представления очень больших целых чисел
- Boolean: представляет логическое значение true или false
- Undefined: представляет одно специальное значение - undefined и указывает, что значение не установлено
- Null: представляет одно специальное значение - null и указывает на отсутствие значения
- Symbol: представляет уникальное значение, которое часто применяется для обращения к свойствам сложных объектов
- Object: представляет комплексный объект

Первые семь типов представляют примитивные типы данных. Последний тип - Object представляет сложный, комплексный тип данных, который состоит из значений примитивных типов или других объектов. Рассмотрим основные примитивные типы данных.

Преобразование может быть явным, когда мы целенаправленно приводим один тип к другому, либо неявным, когда приведение типа происходит автоматически без явных команд.

```
String("123") // Явное преобразование
123 + "" // неявное преобразование
```

В JavaScript преобразование всегда приводит к 3м типам:

- к строке
- к числу
- к логическому значению (true / false)

Приведениекстроке

```
String(null) // "null"
String(undefined) // "undefined"
String(true) // "true"
String(false) // "false"
String(1) // "1"
String(NaN) // "NaN"
String(10000000000 * 900000000000) // "9e+21"
String({}) // "[object Object]"
String({ name: "Ivan" }) // "[object Object]"
String([]) // ""
String([1, 2, 3]) // "1,2,3"
```

В данном примере преобразование происходит очевидным образом.

Тема 5. Объекты языка JS

Внутренний и внешний интерфейс. Геттеры и сеттеры. Функциональное наследование. Защищённые свойства. Перенос свойства в защищённые. Переопределение методов.

Прототип объекта. Свойство F.prototype и создание объектов через new. Встроенные "классы" в JavaScript. Свои классы на прототипах. Наследование классов в JavaScript. Проверка класса: "instanceof". Ошибки, наследование от Error. Примеси.

Объекты JavaScript используются для хранения коллекций различных значений и более сложных сущностей. В JavaScript объекты используются очень часто, это одна из основ языка. Поэтому мы должны понять их, прежде чем углубляться куда-либо ещё.

Объект может быть создан с помощью фигурных скобок {…} с необязательным списком свойств. Свойство – это пара «ключ: значение», где ключ – это строка (также называемая «именем свойства»), а значение может быть чем угодно.

Мы можем представить объект в виде ящика с подписанными папками. Каждый элемент данных хранится в своей папке, на которой написан ключ. По ключу папку легко найти, удалить или добавить в неё что-либо.

При использовании литерального синтаксиса {...} мы сразу можем поместить в объект несколько свойств в виде пар «ключ: значение»:

```
letuser = {     // объект
name: "John",  // под ключом "name" хранится значение "John"
age: 30        // под ключом "age" хранится значение 30
};
```

У каждого свойства есть ключ (также называемый «имя» или «идентификатор»). После имени свойства следует двоеточие ":", и затем указывается значение свойства. Если в объекте несколько свойств, то они перечисляются через запятую.

В объекте user сейчас находятся два свойства:

Первое свойство с именем "name" и значением "John".

Второе свойство с именем "age" и значением 30.

Можно сказать, что наш объект user – это ящик с двумя папками, подписанными «name» и «age».Мы можем в любой момент добавить в него новые папки, удалить папки или прочитать содержимое любой папки.Для обращения к свойствам используется запись «через точку»:

```
// получаем свойства объекта:
alert( user.name ); // John
alert( user.age ); // 30
```

Значение может быть любого типа.

Тема 6. Введение в GoogleAppsScript

Интегрированная среда разработки. Быстрый старт в GoogleAppsScript. Демонстрационный пример. Классы GoogleApps.

Образцы решений - это полнофункциональные проекты AppsScript. Решения решают реальные бизнес-проблемы и демонстрируют, как можно автоматизировать рабочие процессы в GoogleWorkspace. Часто вы можете реализовать решения без необходимости редактирования или обновления кода.

Сервисы - это некие программные классы, у которых есть какие-то методы, свойства, с которыми можно работать и что-то с помощью этих сервисов сделать.По сути, все программирование GoogleAppsScript сводится в задачу взаимодействия с какими-то сервисами.

Чтобы посмотреть с какими сервисами Google мы можем взаимодействовать, можно воспользоваться документацией. Список доступных сервисов находится в разделе Reference.

Здесь мы можем взаимодействовать с G Suite сервисами. Это сервисы, которые предоставляет нам Google. Календарь, контакты, документы, таблицы и.т.д.Со всеми этими сервисами Google мы можем взаимодействовать с помощью GoogleAppsScript. Еще один варианти - это скриптовые сервисы. Это JDBC, для взаимодействия со сторонними базами данных. HTML - для взаимодействия с HTML страницами и.т.д.

Тема 7. Интерфейс DocumentApp

Обзор класса DocumentApp. Получение содержимого документа. Обновление тела документа. Иерархия тела документа. Классы и методы. HTML в документе. Вставка текстового содержимого. Вставка графики и мультимедиа.

Объектная Модель Документа (DOM) – это программный интерфейс (API) для HTML и XML документов. DOM предоставляет структурированное представление документа и определяет то, как эта структура может быть доступна из программ, которые могут изменять содержимое, стиль и структуру документа. Представление DOM состоит из структурированной группы узлов и объектов, которые имеют свойства и методы. По существу, DOM соединяет веб-страницу с языками описания сценариев либо языками программирования.

Веб-страница – это документ. Документ может быть представлен как в окне браузера, так и в самом HTML-коде. В любом случае, это один и тот же документ. DOM предоставляет другой способ представления, хранения и управления этого документа. DOM полностью поддерживает объектно-ориентированное представление веб-страницы, делая возможным её изменение при помощи языка описания сценариев наподобие JavaScript.

Стандарты W3C DOM и WHATWG DOM формируют основы DOM, реализованные в большинстве современных браузеров. Многие браузеры предлагают расширения за пределами данного стандарта, поэтому необходимо проверять работоспособность тех или иных возможностей DOM для каждого конкретного браузера.

Например: стандарт DOM описывает, что метод getElementsByTagName в коде, указанном ниже, должен возвращать список всех элементов <p> в документе.

```
paragraphs = document.getElementsByTagName("P");
// paragraphs[0] этопервый<p>элемент
// paragraphs[1] это второй <p> элемент и т.д.
alert(paragraphs[0].nodeName);
Copy to Clipboard
```

Все свойства, методы и события, доступные для управления и создания новых страниц, организованы в виде объектов. Например, объект document, который представляет сам документ, объект table, который реализует специальный интерфейс DOM HTMLTableElement, необходимый для доступа к HTML-таблицам, и так далее. Данная документация даёт справку об объектах DOM, реализованных Gecko-подобных браузерах.

Небольшой пример вышенаписан на JavaScript, но при этом используется DOM для доступа к документу и его элементам. DOM не является языком программирования, но без него JavaScript не имел бы никакой модели или представления о веб-странице, HTML-документе, XML-документе и их элементах. Каждый элемент в документе - весь документ в целом, заголовок, таблицы внутри документа, заголовки таблицы, текст внутри ячеек таблицы - это части объектной документной модели для этого документа, поэтому все они могут быть доступны и могут изменяться с помощью DOM и скриптового языка наподобие JavaScript.

Вначале JavaScript и DOM были тесно связаны, но впоследствии они развились в различные сущности. Содержимое страницы хранится в DOM и может быть доступно и изменяться с использованием JavaScript, поэтому мы можем записать это в виде приблизительного равенства:

- API (веб либо XML страница) = DOM + JS (язык описания скриптов)
- DOM спроектирован таким образом, чтобы быть независимым от любого конкретного языка программирования, обеспечивая структурное представление документа согласно единому и последовательному API.

Тема 8. ИнтерфейснезависимогоSpreadsheetApp

Сделать данные доступными, полезными и действенными в различных контекстах - одна из основных причин, по которой мы создаем приложения. Как правило, это требует большого количества инженерных работ, но благодаря разработке приложений без кода даже те, у кого нет опыта программирования, могут создавать приложения из самого обычного хранилища данных из всех: электронных таблиц или Google Таблиц.

Например, если сотрудники на местах записывают данные вручную и позже вводят их в GoogleSheets, всю эту работу можно упростить с помощью приложения без кода. Пользователи просто вводили данные в приложение, которое автоматически синхронизировалось с электронными таблицами в облаке. Для многих бизнес-специалистов возможность замены обслуживания электронных таблиц приложением может привести к значительному повышению эффективности. Аналогичным образом, для многих предприятий возможность предоставить нетехническим специалистам возможность самостоятельно создавать такие приложения без значительных ИТ-ресурсов может иметь преобразующее значение, открывая совершенно новые возможности для ускорения инноваций.

Запуск этого процесса может быть таким же простым, как организация электронных таблиц, чтобы платформа приложений без кода, такая как GoogleCloudAppSheet, могла принимать их и создавать полезные прототипы приложений, которые пользователи могут использовать в дальнейшем. Хотя кодирование не требуется, отличные приложения без кода по-прежнему полагаются на хорошо спроектированную структуру данных. В этом посте мы рассмотрим основы подготовки источника данных в GoogleSheets для эффективного создания приложений с использованием AppSheet.

НезависимоеSpreadSheetApp. Создание содержимого и стилей. Динамические данные и формулы. Классы и методы. Связанное SpreadsheetApp. Функции Sheets. Обработка выделенного содержимого.

Тема 9. ИнтерфейсSpreadsheetApp HTML to backend GS code

Хранение всего кода HTML, CSS и JavaScript в одном файле может затруднить чтение и разработку вашего проекта. Хотя AppsScript требует, чтобы клиентский код помещался в файлы .html, вы все равно можете разделить свой CSS и клиентский JavaScript на разные файлы, а затем включить их в основную HTML-страницу с помощью пользовательской функции.

Шаблонный HTML можно использовать для быстрого создания простых интерфейсов, но его использование должно быть ограничено, чтобы обеспечить отзывчивость пользовательского интерфейса. Код в шаблонах выполняется один раз при загрузке страницы, и до завершения обработки клиенту не отправляется никакой контент. Наличие длительных задач в коде скриптлета может привести к замедлению работы пользовательского интерфейса.

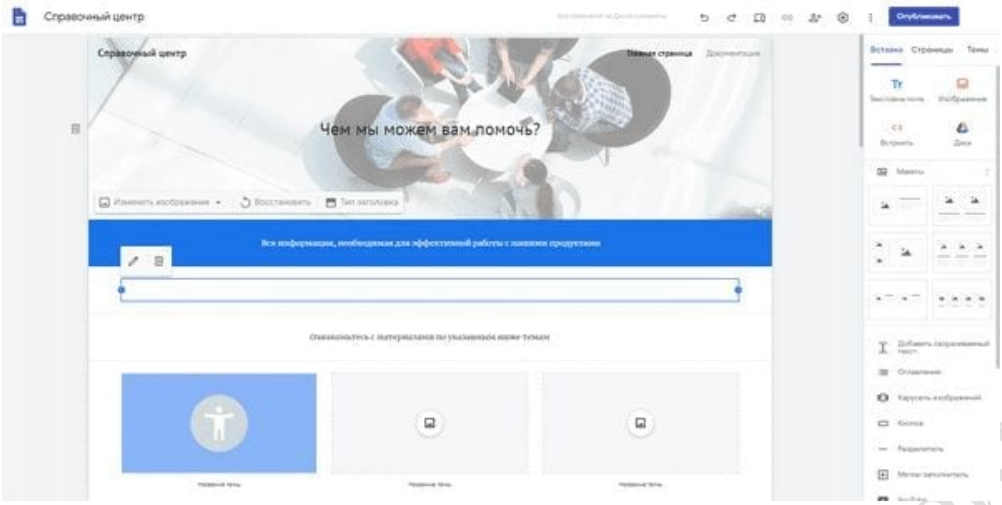
Используйте теги сценариев для быстрых разовых задач, таких как включение другого содержимого или установка статических значений. Все остальные данные должны быть загружены с помощью вызовов google.script.run . Кодирование таким асинхронным способом более сложно, но позволяет пользовательскому интерфейсу загружаться быстрее и дает ему возможность представить пользователю счетчик или другое сообщение о загрузке.

Разработка пользовательского интерфейса. HTMLtobackendGScode. Google Sheet content to Client Side. Triggers OnEditOnChange. Adding new Column. Add Formulas Set Colors. HTML modal to Spreadsheet. Sheet Data to HTML file. Get Sheet Data. Run Script Client Side. Response with Google Sheet Data. Source Code HTML index file.

Тема 10. Интерфейс Sites App

В феврале 2008 года сервис конструктор сайтов GoogleSitesвключили в набор корпоративных приложений GoogleApps.Это весьма качественный сервис (фактически, готовая CMS со встроенным WYSIWYG-редактором и бесплатным хостингом), созданный на базе разработок купленной полтора года назад компании Jotspot.Ранее конструктор сайтов был доступен только для корпоративных заказчиков (конечно, GoogleApps можно было установить на любой домен, но главными пользователями GoogleApps всё-таки являются именно коммерческие организации).

GoogleSites подходит пользователям, которым нужно очень быстро собрать страницу с анонсом мероприятия, презентацией, обучающими материалами, анкетами. Платформа подходит для личных и образовательных целей, коммерческий ресурс на этом конструкторе запустить не получится.



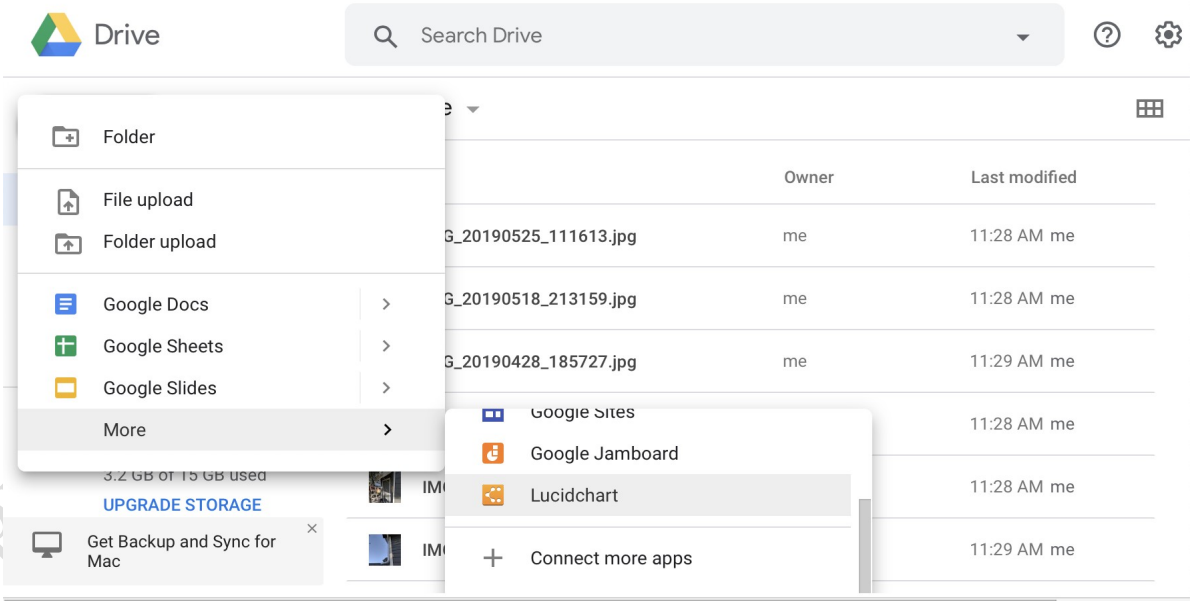
Introduction to Google Sites. New Google Sites. Publish Web App Google Script. Google Site Scripting. Google Script Web App html page.

Тема 11. Интерфейс Drive App

Пользовательский интерфейс (UI) Google Диска – это приложение, предоставляемое Google, в котором пользователи Диска могут создавать, упорядочивать, находить и обмениваться содержимым, хранящимся на Google Диске. Вы можете интегрировать приложение с поддержкой Диска в пользовательский интерфейс Диска, чтобы воспользоваться этими функциями. Вы можете выполнить 2 интеграции:

- С помощью кнопки «Создать» в интерфейсе Диска .
- С помощью пункта меню «Открыть с помощью» пользовательского интерфейса Диска

Если вы хотите, чтобы пользователи DriveUI вызывали ваше приложение для создания файла, интегрируйте свое приложение с кнопкой «Создать» в интерфейсе Drive.



Кнопка «Создать» позволяет пользователям открыть ваше приложение или другие приложения в стиле редактора, такие как GoogleDocs и GoogleSheets, для создания нового документа.

Если вы хотите, чтобы пользователи пользовательского интерфейса Диска могли открывать документы с помощью вашего приложения, интегрируйте свое приложение с пунктом меню «Открыть с помощью» пользовательского интерфейса Диска.

Когда пользователь щелкает правой кнопкой мыши файл в пользовательском интерфейсе Диска, открывается контекстное меню. Контекстное меню содержит пункт

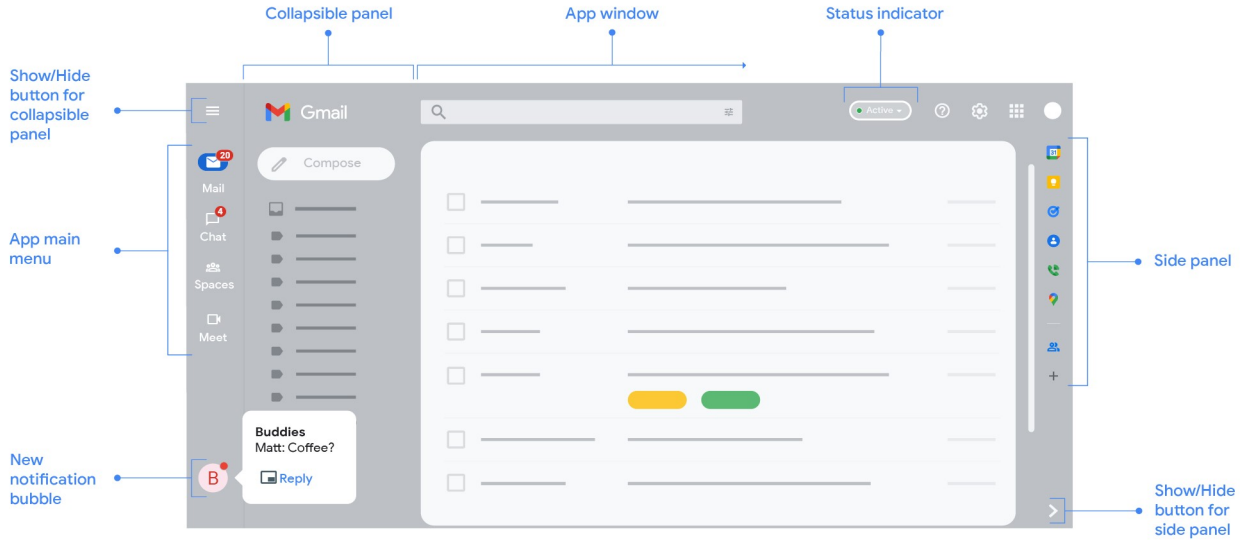
«Открыть с помощью», позволяющий пользователю выбрать приложение для открытия файла.

Google DriveApp Class. Select Folder By ID. Create Folder DriveApp. DriveApp create File. SpreadSheet with DriveApp Data. Create Doc and Move it. DriveApp Search. DriveApp Search Files. DriveAppsearchFoldersParams. Add editors and Delete. Redirect on search string.

Тема 11. Интерфейс Gmail App

В интегрированном интерфейсе Gmail объединены почта, чат, чат-группы и встречи. Теперь естьвозможность:

- видеть все приложения Google, интегрированные в главное меню Gmail;
- открывать меню выбранного приложения в сворачиваемой панели;
- получать всплывающие уведомления о новых сообщениях чата и чат-группы;
- просматривать, что происходит в приложении, не переключаясь на него;
- скрывать или отображать боковую панель.



Почта, чаты, чат-группы и встречи Meet интегрированы в основное меню Gmail. Меню каждого из этих приложений отображается в сворачиваемой панели. Показать или скрыть сворачиваемую панель можно в любой момент.

Introduction to Class GmailApp. Make Draft GmailApp. View your Draft Email. Use HTML Template email. Send out a bunch of Emails. Chat Threads GmailApp. Get Gmail Messages.

Тема 13. Интерфейс Calendar App

Позволяет сценарию читать и обновлять Календарь Google пользователя. Этот класс обеспечивает прямой доступ к календарю пользователя по умолчанию, а также возможность получения дополнительных календарей, которыми владеет пользователь или на которые он подписан.

Характеристики

Свойство	Тип	Описание
Color	Color	Перечисление, представляющее именованные цвета, доступные в службе календаря.
EventColor	EventColor	Перечисление, представляющее цвета именованных событий, доступные в службе календаря.
GuestStatus	GuestStatus	Перечисление, представляющее статусы, которые может иметь гость для события.
Month	Month	Перечисление, представляющее месяцы года.
Visibility	Visibility	Перечисление, представляющее видимость события.
Weekday	Weekday	Перечисление, представляющее дни недели.

Calendar App Service. Add Location Info Calendar App. Calendar Event with options. Full Day events Calendar App. Calendar App Event Series. Calendar App Date Time. Easy way to enter Events. More Calendars. Calendar Settings. Get All Calendars. Calendars By Name. Calendar Events. Calendar Events Explored. Calendar Events into spreadsheet. Google Calendar Events Invites No. Send Weekly Calendar Events.

Тема 14. ПрактическиеисследованияAppsScriptImageUploader

Функция настройки создает папку для хранения всех загруженных файлов и триггер, который срабатывает каждый раз, когда кто-то отправляет форму. Когда пользователь заполняет форму, он выбирает файлы для загрузки и подпапку для хранения файлов. Как только пользователь отправляет форму, скрипт направляет файлы в соответствующую подпапку. Если папка еще не существует, скрипт ее создает.

Это решение использует следующие сервисы:

Служба сценариев - создает триггер, который срабатывает каждый раз, когда кто-то отправляет форму.

Служба свойств - хранит идентификатор триггера, который сценарий создает во время установки, чтобы предотвратить дублирование триггеров.

Служба Диска - во время установки получает расположение формы на Диске и создает папку в том же месте. Когда пользователь отправляет форму, служба Диска направляет файлы в эту папку и, если она выбрана, в назначенную подпапку. Если подпапка еще не существует, сценарий создает ее.

Служба форм - получает файлы и имя папки, выбранные пользователем после отправки формы, и отправляет ее в службу Диска.

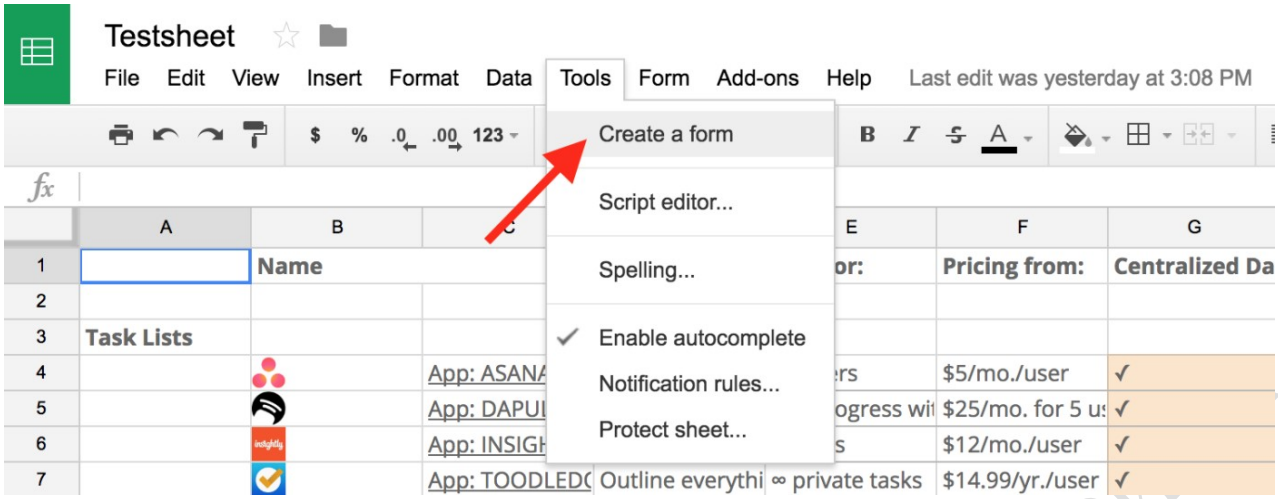
Apps Script Image Uploader project intro. HTML Content Service. HTML Content Template File. HTML Content from File. Create the HTML form. Send Data to Google Script Backend. Send Image to Google Script. Image upload Tweaks. Spreadsheet App tracking uploads. Send Email notification.

Тема 15. ПрактическиеисследованияSetupGoogleForm

GoogleForms является полнофункциональным инструментом для работы с формами, который предоставляется бесплатно вместе с вашей учетной записью Google. Вы можете добавлять стандартные типы вопросов, перетаскивать вопросы в нужном вам порядке, настраивать форму с помощью простых фотографий или цветовых тем, а также собирать ответы в формах или сохранять их в электронной таблице GoogleSheets.

Создание первой формы Google.

Самый простой способ начать создание формы - прямо из приложения GoogleForms. Перейдите на страницу docs.google.com/forms , затем выберите шаблон или создайте пустую форму.



Также есть ссылка на GoogleForms в Документах, Таблицах и Слайдах: нажмите «Файл» > «Создать» > «Форма» , чтобы создать новую пустую форму. Или в Google Таблицах нажмите «Инструменты» > «Создать форму» , чтобы создать пустую новую форму, которая автоматически привязывается к этой электронной таблице. Это самый быстрый способ получить данные в новую или существующую электронную таблицу: откройте электронную таблицу в том месте, где вам нужны данные, запустите форму, и ответы формы будут автоматически сохранены там без каких-либо дополнительных кликов.

В простых контактных формах требуется всего несколько полей, но более длинные опросы могут быстро переполниться десятками вопросов на одной странице. Вот где пригодятся разделы: они позволяют разбить форму на куски, чтобы отвечать на один набор вопросов за раз.

Просто нажмите последнюю кнопку на правой панели инструментов, чтобы добавить раздел под текущим вопросом. Каждый раздел имеет собственное название и описание, а также кнопку со стрелкой вверху, позволяющую отображать или скрывать вопросы и поддерживать порядок в редакторе форм.

Вы можете перетаскивать вопросы между разделами, но не можете менять порядок целых разделов. Вместо этого вы можете переместить вопросы, а затем удалить этот раздел. Или, если вы хотите повторно использовать раздел, просто щелкните меню раздела и нажмите « Дублировать раздел» , чтобы получить еще одну копию этих вопросов.

Вот некоторые из лучших надстроек для Форм, с которых можно начатьинтеграцию:

- CheckItOut позволяет вам возвращать или удалять элементы с помощью формы, по сути, путем перестановки данных из одной категории в другую в электронной таблице. Это отличный инструмент для управления инвентарем или общими элементами, или его можно творчески использовать, например, для утверждения задач или выполнения других заданий, где вам нужно перемещать элементы между двумя категориями.

- ChoiceEliminatorLite исключает варианты из вопросов с несколькими вариантами ответов, списка или флажка, если они уже были выбраны. Это отличный способ, скажем, создать форму регистрации, в которой каждый респондент может выбрать один день, или форму заказа товаров в ограниченном количестве.

- Директор данных добавляет ответы форм на альтернативные листы и отправляет уведомления по электронной почте в зависимости от условий. Вы можете использовать его для автоматической сортировки всех похожих записей на разные листы.

- docAppender добавляет результаты формы в конец документа GoogleDocs, а не в электронную таблицу. Каждый ответ может быть добавлен в уникальные документы на основе вопросов формы, или каждый из них может быть добавлен в один и тот же документ.

FormNotifications отправляет вам настраиваемые уведомления по электронной почте и, при необходимости, формирует респондентов с подробными сведениями о результатах формы и сообщением с благодарностью.

- FormPublisher создает шаблоны документов GoogleDocs, PDF-файлов или уникальных электронных таблиц для каждой записи, а затем отправляет их по электронной почте.

- formRecycler импортирует вопросы из других форм, чтобы быстро использовать их повторно, не копируя всю форму.

- Setup Google Form. Spreadsheet Bound Script. Google Script Setup Project triggers. Setup Email to Send. MailApp Send Email. Email HTML template. HTML Service replace content. HTML to PDF and Attach it. Send automatic emails. iterate Sheet Data Send Emails. Review and update code for project.

Тема 16. Интеграция приложений

Интеграция в GoogleApps достигла того, что документы, которые пришли вам по почте, уже сразу из почтового ящика можно открыть для работы GoogleDocs&Spreadsheets. Переключение между различными сервисами осуществляется простым кликом мышки. Впрочем, нет проблем в том, чтобы держать их все открытыми и готовыми для работы в разных окнах браузера.

Таким образом, GoogleApps создает собой среду, которая удобна не только для совместной работы нескольких человек, которые находятся в разных местах, но и для организации совместной работы большого числа сотрудников, которые могут быть раскиданы по различным странам. Учитывая то, что компания Google довольно часто предоставляет API к своим продуктам, тем самым позволяя создавать собственные приложения, которые еще более интегрируют приложения Google в конкретную среду работы, можно сказать, что GoogleApps может стать очень удобной средой для совместной работы. Кроме того, компания Google постоянно ведет работу над улучшением существующих сервисов и добавлением новых. В ближайших планах добавление сервиса для миграции с различных почтовых клиентов на GMail и возможность создавать и редактировать презентации, то есть сервис, аналогичный программе PowerPoint.

GoogleAppEngine позволяет выполнять ваши веб-приложения в инфраструктуре Google. Приложения AppEngine легко создавать, поддерживать и усовершенствовать по мере увеличения трафика и хранилища данных. При работе с AppEngine вам не понадобится поддерживать сервер: просто загрузите свое приложение, и пользователи смогут работать с ним.Приложение можно опубликовать в собственном домене (например, http://www.example.com) с помощью Служб Google. Или воспользоваться бесплатным именем в домене appspot.com. Приложение можно сделать доступным для всех или предоставить доступ только участникам вашего коллектива.

GoogleAppEngine поддерживает приложения, написанные на нескольких языках программирования. Благодаря среде выполнения JavaAppEngine можно создавать приложения с помощью стандартных технологий Java, в том числе JVM, сервлетовJava и языка программирования Java, или другого языка, использующего интерпретатор или компилятор на JVM, напримерJavaScript или Ruby. Кроме того, AppEngine предоставляет специальную среду выполнения Python, которая включает быстрый интерпретатор и стандартную библиотеку Python. Среды выполнения Java и Python разработаны специально для того, чтобы приложения могли быстро и безопасно выполняться без взаимодействия с другими приложениями в системе.

ContactFormSetup. Create Spreadsheet and setup. Sending an Email with Apps Script. Google Script send email. Spreadsheet App add content. Test Sheet Google Script. Google Script Trigger Examples. Get Data Parameters. JavaScript Fetch. Outdate HTML in Submit. Google Script Application. Send email on submit. Random ID function and Date Add. Emailtemplate. CreateWebApp.

Пример презентационного материала для проведения занятия

JavaScript

JavaScript – мультипарадигменный язык программирования, который используется для создания сценариев, выполняющихся в веб браузерах.

JavaScript является реализацией спецификации **ECMAScript** (стандарт ECMA-262).

В 1995 году Бренден Эйх начал работать над новым скриптовым языком для браузера Netscape Navigator. Также в разработке участвовали сооснователь Netscape Communications Марк Андрессен и сооснователь Sun Microsystems Билл Джой.


Брендан Эйх


Марк Андрессен


Билл Джой

Редакторы кода

IDE (Integrated Development Environment) – комплекс инструментов для разработки программного обеспечения.

Visual Studio Community
https://visualstudio.com/
Бесплатно

Web Storm
https://www.jetbrains.com/webstorm/
Платно

Редакторы кода

Visual Studio Code
https://code.visualstudio.com/

NotePad++
https://notepad-plus-plus.org/

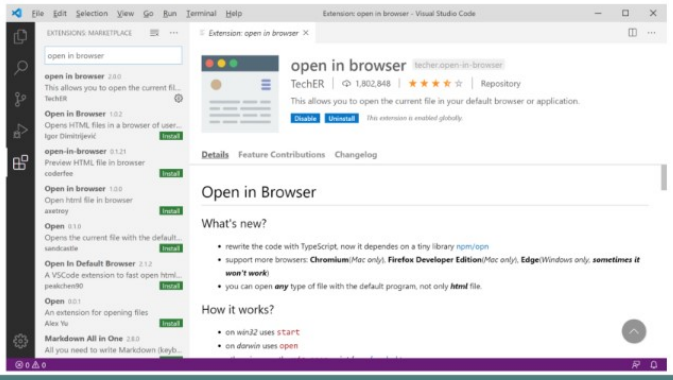
Atom
https://atom.io/

Sublime Text
http://www.sublimetext.com/

Ex.indexEx1

Open in browser

open in browser – расширение для Visual Studio Code, позволяющее открывать текущий файл в браузере (с помощью контекстного меню или комбинации клавиш Alt + B).



Структура кода

Инструкция – синтаксическая единица языка программирования, выражающая действие, выполняемое программным кодом.

Приложение состоит из набора последовательных инструкций. Инструкции могут содержать вложенные компоненты, например, выражения.

Инструкции отделяются точкой с запятой **;**

```
let div = document.querySelector("div"); div.innerText = "Привет мир!";
```

Обычно каждая инструкция находится на отдельной строке.

```
let div = document.querySelector("div");
div.innerText = "Привет мир!";
```

В большинстве ситуаций новая строка подразумевает точку с запятой, но лучше использовать явно **;**

ЗВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

Основы протокола HTTP

1. Клиент и сервер.
2. Полная форма URL.
3. Формат запроса клиента.
4. Формат ответа сервера.

Понятие и обзор облачных сервисов

5. Определение облачного хранилища и сервиса.
6. Виды облачных сервисов.
7. GoogleDrive, MicrosoftOneDrive, Dropbox, Яндекс.Диск.

Обзор языка JS

8. Переменные. Операции. Логические операторы.
9. Операторы цикла. Перехват ошибок.
10. Локальные переменные. Внешние переменные.
11. Аргументы по умолчанию. Возврат значения.
12. Функциональные выражения.

Обзор языка JS

13. Объекты как ассоциативные массивы.
14. Перебор свойств.
15. Копирование по значению и ссылке. Клонирование объектов.
16. Глобальный объект.
17. Локальные переменные для объекта.
18. Модули через замыкания. Управление памятью.

Обзор языка JS

19. Внутренний и внешний интерфейс.
20. Геттеры и сеттеры. Функциональное наследование.
21. Защищённые свойства.
22. Прототип объекта.
23. Проверка класса: "instanceof".
24. Ошибки, наследование от Error. Примеси.

Введение в GoogleApps Script

25. Интегрированная среда разработки.
26. Демонстрационный пример.
27. Классы GoogleApps.

Интерфейс DocumentApp

28. Обзор класса DocumentApp.
29. Получение содержимого документа.
30. Вставка текстового содержимого.
31. Вставка графики и мультимедиа.

Интерфейс SpreadsheetApp

32. НезависимоеSpreadSheetApp.
33. Создание содержимого и стилей.
34. Динамические данные и формулы.
35. Классы и методы.

Интерфейс SpreadsheetApp

36. Разработка пользовательского интерфейса.
37. HTMLtobackendGScode. Google
38. Add Formulas Set Colors.
39. Get Sheet Data.
40. Run Script Client Side.

Интерфейс SitesApp

41. Introduction to Google Sites.
42. Publish Web App Google Script.
43. Google Site Scripting.

Интерфейс DriveApp

44. Google DriveApp Class.
45. Select Folder By ID.
46. SpreadSheet with DriveApp Data.
47. DriveApp search. Folders Params. Add editors and Delete. Redirect on search string.

Интерфейс Gmail App

48. Introduction to Class GmailApp.
49. Make Draft GmailApp.
50. Template email.
51. Chat Threads GmailApp.
52. Get Gmail Messages.

Интерфейс CalendarApp

53. Calendar App Service.
54. Add Location Info Calendar App.
55. Full Day events Calendar App.
56. Calendar Events Explored.

Практические исследования

57. Apps Script Image Uploader project intro.
58. HTML Content Template File.
59. Send Data to Google Script Backend.
60. Send Image to Google Script.
61. Image upload Tweaks.

Практические исследования

62. Google Script Setup Project triggers.
63. HTML Service replace content.
64. HTML to PDF and Attach it.
65. Review and update code for project.

Интеграция приложений

66. ContactFormSetup.
67. Create Spreadsheet and setup.
68. Sending an Email with Apps Script.
69. Google Script send email.

4 ЗАДАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ

Задание к 5 лекции

Задание: Необходимо создать автономный и контейнерный скрипты согласно инструкции. Проверить корректность их работы.

Задание к 6 лекции

Задание: Используя справочную информацию в прикрепленном файле и документацию на официальном источнике, создать 3 кейса автоматизации сервиса GoooglеSpreadsheet. Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины таблиц.

Задание к лекции 7

Задание: Используя документацию на официальном источнике, создать 3 кейса автоматизации сервиса GoooglеDocument. Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины документов.

Задание к 8 лекции

Задание: Используя документацию на официальном источнике, создать 3 кейса автоматизации сервиса SitesApp . Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины документов.

Задание к 9 лекции

Задание: Используя документацию на официальном источнике, создать 3 кейса автоматизации сервиса DriveApp. Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины документов.

Задание к 10 лекции

Задание: Используя документацию на официальном источнике, создать 3 кейса автоматизации сервиса GmailService. Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины документов.

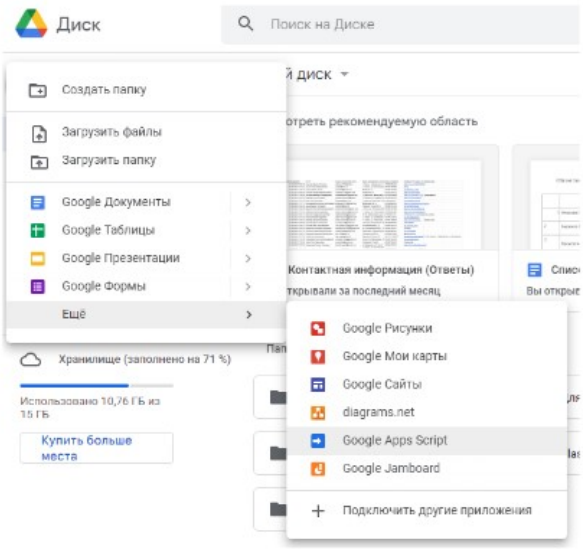
Задание к лекции 11

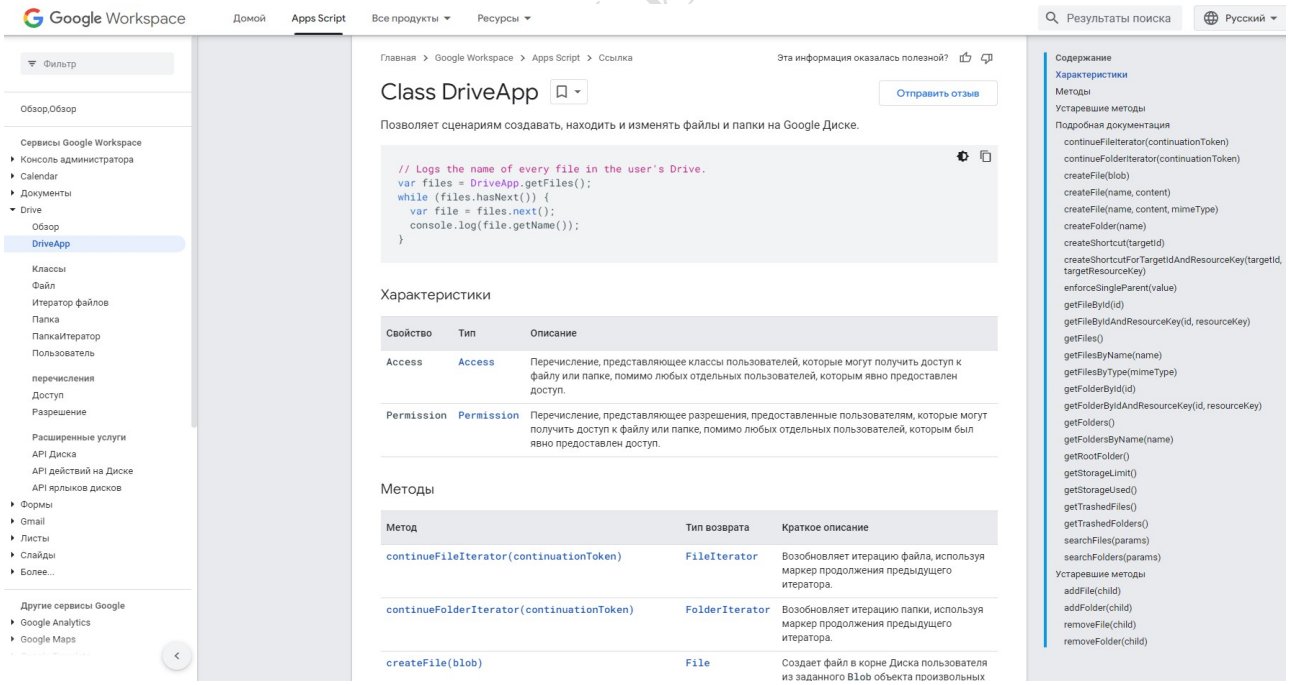
Задание: Используя документацию на официальном источнике, создать 3 кейса автоматизации сервиса CalendarApp. Оформить отчет, в котором описать кейсы, приложить листинги скриптов и скрины документов.

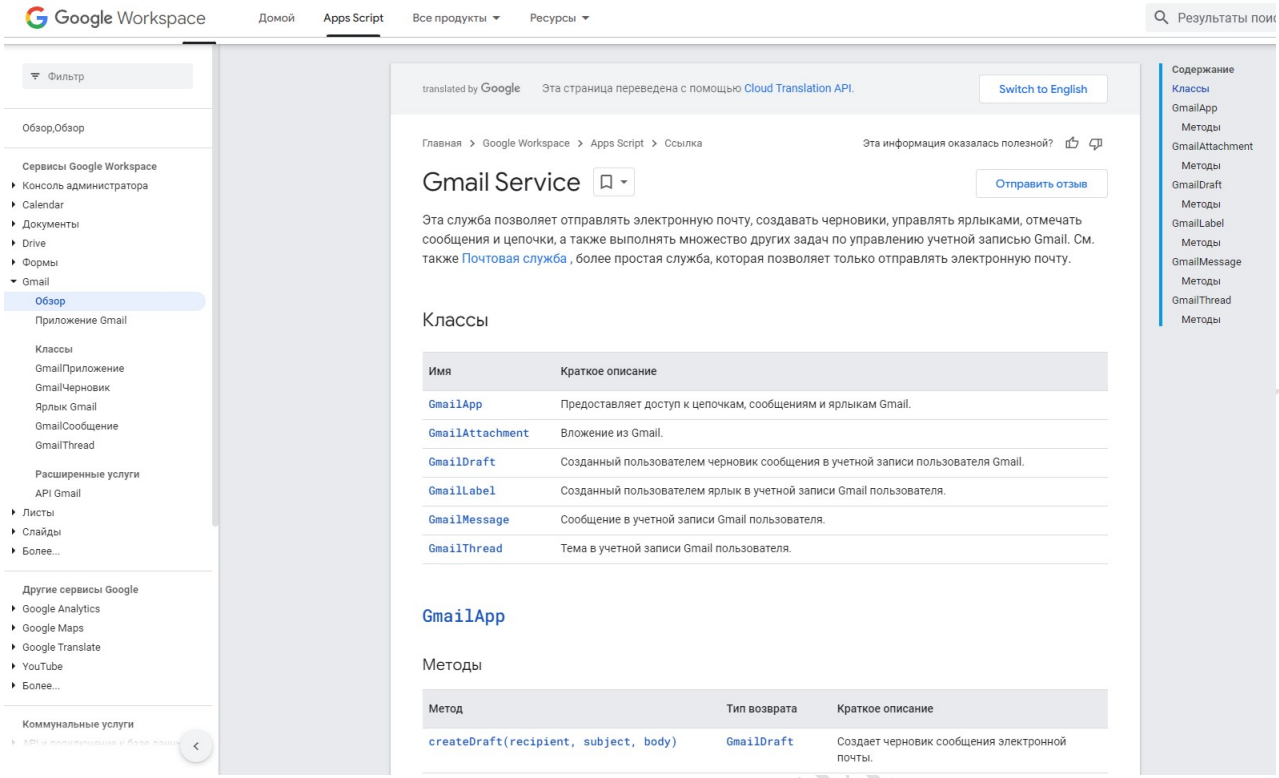
Примеры материалов заданий для работы

Чтобы построить автоматизацию, выполните следующие действия:

1. Войдите в свою учетную запись Google.
2. Существует 2 способа войти в редактор скриптов:
   1 способ - перейти по ссылке script.google.com.
   2 способ – войти в гугл диск – нажать кнопку Создать – выбрать Google Apps Script







Google Apps Script: основные команды для автоматизации Гугл Таблиц

Навигация по классу SpreadsheetApp.

Spreadsheet (таблица как документ)

```
SpreadsheetApp.openByUrl (url)
SpreadsheetApp.openById (id)
SpreadsheetApp.getActive()
SpreadsheetApp.getActiveSpreadsheet()
SpreadsheetApp.open(file)
```

Вспомогательные функции:
ss.getUrl() ss.getId()
ss.getName() ss.getFormUrl()

Sheet (лист)

```
ss.getSheetByName(name)
ss.getActiveSheet()
```

Вспомогательные функции:
sheet.getName() =sheet.getSheetName()
sheet.getSheetId() range.getSheet()
Получить все листы таблицы: ss.getSheets()
sheet.getIndex() sheet.getFormUrl()

Range (диапазон)

```
getRange
(row,column,numRows,numColumns)
getRange ( a1Notation )
getActiveRange()
getRange (row, column, numRows)
```

Вспомогательные функции:

```
range.getA1Notation()  range.getRow()  range.getColumn()
 sheet.getLastRow()  sheet.getLastColumn()
range.getNumRows() range.getNumColumns()
range.offset(rowOffset, columnOffset)
```

Cell (ячейка)

```
getRange(row,
column)
getRange(a1Notation)
getCurrentCell ( )
getActiveCell ( )
```

Основные команды:

ссылка на официальную документацию по Google Apps Script

Создать

Spreadsheet (таблица как документ)

```
SpreadsheetApp.create (name)
```

Вставить лист

```
ss.insertSheet ( )
ss.insertSheet ( sheetIndex )
ss.insertSheet ( sheetName )
ss.insertSheet ( sheetName, sheetIndex )
```

Добавить строку с контентом

```
sheet.appendRow(rowContent)
```

Вставить строки

```
sheet.insertRowBefore ( beforePosition )
sheet.insertRowAfter ( afterPosition )
sheet.insertRows ( rowIndex )
sheet.insertRows ( rowIndex, numRows )
sheet.insertRowsBefore(beforePosition,howMany)
sheet.insertRowsAfter ( afterPosition, howMany )
```

Вставить столбцы

```
sheet.insertColumnBefore(beforePosition)
sheet.insertColumnAfter(afterPosition)
sheet.insertColumns(columnIndex)
sheet.insertColumns(columnIndex,numColumns)
sheet.insertColumnsBefore(beforePosition,howMany)
sheet.insertColumnsAfter(afterPosition,howMany)
```

Копировать

Spreadsheet (таблица как документ)

```
ss.copy ( name)
```

Лист

```
ss.duplicateActiveSheet()
sheet.copyToSpreadsheet(spreadsheet )
```

Добавить строку с контентом

```
range.copyTo ( destination )
range.copyTo(destination,copyPasteType,
transposed)
range.copyTo ( destination, options )
```

Переименовать

Таблицу	Лист
ss.rename (name)	sheet.setName (name)

Получить/установить значение

```
getValue() - setValue()
getValues() - setValues()
range.getNote - range.setNote (примечания)
activate () - активировать
```

Удалить

Лист

```
ss.duplicateActiveSheet ( )
sheet.copyToSpreadsheet ( spreadsheet )
```

Удалить строки и столбцы

Одну строку sheet.deleteRow(rowPosition)
Несколько:sheet.deleteRows(rowPosition, howMany)
Один столбец:
sheet.deleteColumn(columnPosition)
Несколько столбцов:
sheet.deleteColumns (columnPosition,howMany)
Диапазон ячеек range.deleteCells (shiftDimension)

Очистить

Лист

```
sheet.clear ( )
sheet.clearContents ( )
sheet.clearFormats ( )
sheet.clearNotes ( )
sheet.clearConditionalFormatRules()
```

Диапазон

```
range.clear ( ) range.clearContent ( )
range.clearFormat ( )
range.clearNote ( )
range.clearDataValidations ( )
```

Триггеры

Простые триггеры (на редактирование, пример)

```
function onEdit(e) {
var range = e.range,
    row =  range.getRow(),
    column = range.getColumn();
range.offset(0, 1).setValue(new Date().toLocaleString("ru"))}
Перечень объектов: e.range,e.value,e.sourse
```

Простые триггеры (на открытие, пример "Создание пользовательского меню")

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu('МОЁ МЕНЮ')
    .addItem('Запустить функцию 1', 'myFunction1')
    .addToUi();}
```

Кнопки

Вставка → Изображение → Изображение поверх ячеек
или Вставка → Рисунок (нарисовать любую фигуру)
дальше одинаково:
• • • → Назначить скрипт → Вписать **название функции без ( )**

Управление триггерами вручную

Проект Apps Script → Слева "Триггеры" →
В правом нижнем углу нажмите Добавить триггер →
Выберите и настройте тип триггера → Сохранить

Программное управление триггерами

```
function createTimeDrivenTrigger() {
  ScriptApp.newTrigger('myFunction')
    .timeBased() .onWeekDay(ScriptApp.WeekDay.MONDAY)
    .atHour(9) .create();}
```

Пользовательские функции (пример)

```
function kilometersToNauticalMiles ( kilometers ) {
  var nauticalMiles = kilometers/1.8522;
  return nauticalMiles }
```

Скрипт можно записать с помощью макроса: Инструменты → Макросы → Запись макроса

ДРУГИЕ ПОЛЕЗНЫЕ ФУНКЦИИ:
Отправить письмо: MailApp.sendEmail(recipient, subject, body)
Отправить событие в календарь: CalendarApp.getCalendarById("ID").createEvent(title, startTime, endTime)
Поисковый запрос: ss.createTextFinder(findText).findNext()
Перевести текст: LanguageApp.translate(text, sourceLanguage, targetLanguage)
Вывести окно с сообщением:SpreadsheetApp.getUi().alert('message')
Получить email пользователя сессии: Session.getActiveUser().getEmail()
Получить текущую дату и время: new Date().toLocaleString('ru')
Узнать, что в переменной: Logger.log ('переменная: '+ x + ', тип: '+ typeof x)

5 ТЕСТОВЫЕ ЗАДАНИЯ (примеры)

1. What does the Google DriveApp Class allow you to do?

Выберите один ответ.

- Allows scripts only to create files and folders.
- Allows scripts only to modify files and folders.
- Class does nothing, it is just definition.
- Class is absent in Google DriveApp.
- Allows scripts to create, find, and modify files and folders in Google Drive.

12. What command is used to import data from another Google spreadsheet?

Выберите один ответ.

- ImportXML
- ImportDoc
- ImportHTML
- ImportTable
- ImportRange

85. What is used to append an existing table row?

Выберите один ответ.

- getCell(rowIndex, cellIndex)
- appendTableRow(tableRow)
- appendTableRow()
- getColSpan()
- appendTable()

84. What does appendImage(image) do?

Выберите один ответ.

- Retrieves all the InlineImages contained in the section.
- Determines whether the element is at the end of the Document.
- Creates and inserts an InlineImage from the specified image blob, at the specified index.
- Creates and appends a new InlineImage from the specified image blob.
- Retrieves the element's attributes.

31. How to find files DriveApp?

Выберите один ответ.

- searchFiles(params)
- findFile(params)
- findFiles(params)
- lookForFiles(params)
- searchFile(params)

30. What type of functions does not exist in Sheets?

Выберите один ответ.

- Web function
- Logic function
- Information functions
- Style function
- Engineering functions

21. What is the definition of a GmailApp class?

Выберите один ответ.

- Class of a user-created label in a user's Gmail account.
- Class of a user-created draft message in a user's Gmail account.
- Class that provides access to Gmail threads, messages, and labels.
- Class that accesses and modifies Google Sheets Files.
- Class that creates and accesses Google Sites.

19. What represents argument of doGet(e) and doPost(e) functions?

Выберите один ответ.

- Argument represents an event parameter that can contain information about any request and response parameters.
- Argument represents an event parameter that can contain information about any response parameters.
- Argument represents an event parameter that can contain information about any request parameters.
- Argument represents array of parameters of the response.
- Argument represents an event parameter that can contain information in request body.

13. What command imports tables and lists from a web page?

Выберите один ответ.

- ImportXML
- ImportHTML
- ImportTable
- ImportRange
- ImportDoc

72. What we can put into parameter "options" for method createDraft(...)?

Выберите один ответ.

- the subject line
- an array of files to send with the email
- the body of the email
- the addresses of the recipient
- a JavaScript object that specifies advanced parameters

49. How to create Calendar Events?

Выберите один ответ.

- getEventSeries().
- events.update().
- events.insert() .
- setEventSeries().

10. What type of links does Spreadsheet have?

Выберите один ответ.

- Mixed links
- Universal links
- Special links
- Personalized links
- Simple links

63. Which trigger runs when a user opens a spreadsheet, document, presentation, or form that the user has permission to edit?

Выберите один ответ.

- doPost
- onOpen
- doGet
- onInstall
- onEdit

11. When does the "on Edit" trigger fire?

Выберите один ответ.

- Automatically
- Is always
- When user changes a value in a spreadsheet
- Randomly
- When user does not make changes to the spreadsheet

55. How to add a paragraph with style?

Выберите один ответ.

- var par = body.appendParagraph('A bold, italicized paragraph.');
- var par = body.newParagraph('A bold, italicized paragraph.');
- var par = body.addParagraph('A bold, italicized paragraph.');
- var par = body.newAppendParagraph('A bold, italicized paragraph.');
- var par = body.paragraph('A bold, italicized paragraph.');

87. Which of the following is not a cloud service?

Выберите один ответ.

- Dropbox
- Microsoft OneDrive
- Яндекс.Диск
- Google Drive
- GitHub

29. Which function relates to engineering functions?

Выберите один ответ.

- IMCOT
- TYPE
- HYPERLINK
- IFNA
- ISEMAIL

59. What function does this Google Sheets interface element perform?

Выберите один ответ.

- Vertical text alignment
- Borders
- Border Style
- Merge cells
- Horizontal text alignment

60. For what purposes is the DataExecutionState class used?

Выберите один ответ.

- An enumeration of possible special paste types.
- An enumeration of data source parameter types.
- An enumeration of data execution error codes.
- An enumeration of data source types.
- An enumeration of data execution states.

83. Which class is not contained in the Document?

Выберите один ответ.

- Body
- Text
- Bookmark
- DurationItem
- Position

14. What command is used to import other data, not tables and lists from a web page?

Выберите один ответ.

- ImportDoc
- ImportPabe
- ImportXML
- ImportRange
- ImportHTML

43. What method we need to use to send a message with GmailApp?

Выберите один ответ.

- getMessageById(id)
- createLabel(name)
- refreshMessage(message)
- starMessage(message)
- sendEmail(recipient, subject, body, options)

33. How to find folders DriveApp?

Выберите один ответ.

- findFolder(params)
- searchFolder(params)
- searchFolders(params)
- findFolders(params)
- lookForFolders(params)

34. What type of data is returned by the methods used to search for files and folders DriveApp?

Выберите один ответ.

- File and FolderIterator
- Success message
- FileIterator and FolderIterator
- File and Folder
- FileIterator and Folder

8. What language is used in the Google Sheet app?

Выберите один ответ.

- GoLang
- Java
- C#
- JavaScript
- Google Apps Script

Учреждение образования
«Гомельский государственный университет имени Франциска Скорины»

УТВЕРЖДАЮ
Проректор по учебной работе
ГГУ имени Ф. Скорины
_____ И.В. Семченко
22.11.2019
(дата утверждения)
Регистрационный № УД-31-2019-643/ уч.

ПРОГРАММИРОВАНИЕ ОБЛАЧНЫХ СЕРВИСОВ

Учебная программа учреждения высшего образования по учебной дисциплине
для специальности 1-45 80 01 Системы и сети инфокоммуникаций
Профилизации Виртуализация сетевых сред

2019 г.

Учебная программа составлена на основе: образовательного стандарта ОСВО 1-45 80 01-2019 и учебного плана по специальности высшего образования второй ступени (магистратура) 1-45 80 01 Системы и сети инфокоммуникаций регистрационный № № I 45-2-01/Д-19, I 45-2-01/З-19, утв. 09.04.2019.

СОСТАВИТЕЛЬ:
В.Д. Левчук, заведующий кафедрой АСОИ

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой автоматизированных систем обработки информации
(протокол № _9__ от _16.04.2019);

Научно-методическим советом Учреждения образования «Гомельский государственный университет имени Франциска Скорины».
(протокол № _8__ от _17.05.2019);

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Дисциплина «Программирование облачных сервисов» специальности 1-45 80 01 Системы и сети инфокоммуникаций является дисциплиной компонента УВО модуля «Организация взаимодействия в распределенных системах» и изучается магистрантами первого года обучения.

Актуальность изучения дисциплины связана с применением и эксплуатацией облачных ресурсов, программируемого доступа к ним, обеспечением электронного документооборота на предприятиях.

Необходимость дисциплины «Программирование облачных сервисов» обусловлена требованиями учебного плана по специальности 1-45 80 01 Системы и сети инфокоммуникаций.

ЦЕЛЬ, ЗАДАЧИ, РОЛЬ УЧЕБНОЙ ДИСЦИПЛИНЫ

Целью дисциплины «Программирование облачных сервисов» является овладение основами программируемого доступа к облачным сервисам и ресурсам.

Задачами дисциплины являются:

– изучение теоретических основ создания приложений для облачных сервисов;

– получение практических навыков создания приложений для облачных сервисов.

В результате изучения дисциплины магистрант должен:

знать:

– теоретические аспекты программирования облачных сервисов;

– ключевые понятия архитектуры программного обеспечения на стороне клиента и сервера;

уметь:

– использовать теоретические аспекты программирования web-приложений для их реализации;

владеть:

– прикладным программным обеспечением для настройки и развертывания приложений в облачном сервисе.

ТРЕБОВАНИЯ К УРОВНЮ ОСВОЕНИЯ СОДЕРЖАНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

В результате изучения учебной дисциплины «Программирование облачных сервисов» формируются следующие компетенции:

СК-2 Выполнять научно-исследовательские и опытно-конструкторские работы в области систем и сетей инфокоммуникаций

СК-4Владеть технологиями проектирования и разработки облачных структур в локальных сетях.

МЕТОДЫ (ТЕХНОЛОГИИ) ОБУЧЕНИЯ

Основными методами (технологии) обучения являются:

– словесные, наглядные, практические (по источнику изложения учебного материала);

– репродуктивные, объяснительно-иллюстрированные, поисковые, исследовательские, проблемные и др. (по характеру учебно-познавательной деятельности);

– индуктивные и дедуктивные (по логике изложения и восприятия учебного материала).

ОРГАНИЗАЦИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ МАГИСТРАНТОВ

При изучении учебной дисциплины рекомендуется использовать следующие формы самостоятельной работы:

– изучение учебной литературы;

– подготовка к лабораторным и практическим работам;

– решение индивидуальных задач в аудитории во время проведения лабораторных занятий под контролем преподавателя;

– реализация проектов средствами интегрированных средств разработки приложений для облачных сервисов.

ДИАГНОСТИКА КОМПЕТЕНЦИИ МАГИСТРАНТА

Учебным планом специальности в качестве формы итогового контроля по дисциплине «Программирование облачных сервисов» предусмотрен зачет.

Для текущего контроля и самоконтроля знаний и умений студентов по данной дисциплине используется: выполнение лабораторных работ с их защитой.

Дисциплина компонента УВО «Программирование облачных сервисов» изучается магистрантами 1 года обучения (1семестр) дневной формы обучения и 1 года обучения (1 семестр) заочной формы обучения для специальности: 1-45 80 01 Системы и сети инфокоммуникаций.

Общее количество часов – 230, зачетных единиц – 6.

Дневная форма обучения: аудиторное количество часов – 72; из них: лекционных занятий – 32, практических занятий – 24, лабораторных работ – 16, (управляемая самостоятельная работа – 14).

Форма отчётности – зачет.

Заочная форма обучения: аудиторное количество часов – 16; из них: лекционных занятий – 6, практических занятий – 4, лабораторных работ – 6.

Форма отчётности – зачет, зачетных единиц – 6.

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Тема 1. Основы протокола HTTP

Клиент и сервер. Полная форма URL. Формат запроса клиента. Методы. Формат ответа сервера.

Тема 2. Понятие и обзор облачных сервисов

Определение облачного хранилища и сервиса. Виды облачных сервисов. GoogleDrive, MicrosoftOneDrive, Dropbox, Яндекс.Диск.

Тема 3. Обзор языка JS

Переменные. Операции. Логические операторы. Операторы цикла. Перехват ошибок.

Объявление. Локальные переменные. Внешние переменные. Параметры. Аргументы по умолчанию. Возврат значения.Функциональные выражения.

Тема 4. Обзор языка JS

Объекты как ассоциативные массивы. Операции с объектом. Перебор свойств. Передача по ссылке. Копирование по значению и ссылке. Клонирование объектов.

Глобальный объект. Замыкания, функции изнутри. [[Scope]] для newFunction. Локальные переменные для объекта. Модули через замыкания. Управление памятью.

Тема 5. Обзор языка JS

Внутренний и внешний интерфейс. Геттеры и сеттеры. Функциональное наследование. Защищённые свойства. Перенос свойства в защищённые. Переопределение методов.

Прототип объекта. Свойство F.prototype и создание объектов через new. Встроенные "классы" в JavaScript. Свои классы на прототипах. Наследование классов в JavaScript. Проверка класса: "instanceof". Ошибки, наследование от Error. Примеси.

Тема 6. ВведениевGoogleAppsScript

Интегрированная среда разработки. Быстрый старт вGoogleAppsScript. Демонстрационный пример. Классы GoogleApps.

Тема 7. ИнтерфейсDocumentApp

ОбзорклассаDocumentApp. Полученисодержимогодокумента. Обновлениетеладокумента. Иерархиятеладокумента. Классыиметоды. HTMLвдокументе. Вставкатекстовогосодержимого. Вставкаграфикиимультимедиа.

Тема 8. ИнтерфейсSpreadsheetApp

НезависимоеSpreadSheetApp. Созданиесодержимогоистилей. Динамические данные и формулы. Классы и методы. СвязанноеSpreadsheetApp. ФункцииSheets. Обработка выделенного содержимого.

Тема 9. ИнтерфейсSpreadsheetApp

Разработкапользовательскогоинтерфейса. HTML to backend GS code. Google Sheet content to Client Side. Triggers OnEditOnChange. Adding new Column. Add Formulas Set Colors. HTML modal to Spreadsheet. Sheet Data to HTML file. Get Sheet Data. Run Script Client Side. Response with Google Sheet Data. Source Code HTML index file.

Тема 10. ИнтерфейсSitesApp

Introduction to Google Sites. New Google Sites. Publish Web App Google Script. Google Site Scripting. Google Script Web App html page.

Тема 11. Интерфейс DriveApp

Google DriveApp Class. Select Folder By ID. Create Folder DriveApp. DriveApp create File. SpreadSheet with DriveApp Data. Create Doc and Move it. DriveApp Search. DriveApp Search Files. DriveAppsearchFoldersParams. Add editors and Delete. Redirect on search string.

Тема 11. ИнтерфейсGmailApp

Introduction to Class GmailApp. Make Draft GmailApp. View your Draft Email. Use HTML Template email. Send out a bunch of Emails. Chat Threads GmailApp. Get Gmail Messages.

Тема 13. ИнтерфейсCalendar App

Calendar App Service. Add Location Info Calendar App. Calendar Event with options. Full Day events Calendar App. Calendar App Event Series. Calendar App Date Time. Easy way to enter Events. More Calendars. Calendar Settings. Get All Calendars. Calendars By Name. Calendar Events. Calendar Events Explored. Calendar Events into spreadsheet. Google Calendar Events Invites No. Send Weekly Calendar Events.

Тема 14. Практическиеисследования

Apps Script Image Uploader project intro. HTML Content Service. HTML Content Template File. HTML Content from File. Create the HTML form. Send Data to Google Script Backend. Send Image to Google Script. Image upload Tweaks. Spreadsheet App tracking uploads. Send Email notification.

Тема 15. Практическиеисследования

Setup Google Form. Spreadsheet Bound Script. Google Script Setup Project triggers. Setup Email to Send. MailApp Send Email. Email HTML template.

HTML Service replace content. HTML to PDF and Attach it. Send automatic emails. iterate Sheet Data Send Emails. Review and update code for project.

Тема 16. Интеграцияприложений

ContactFormSetup. Create Spreadsheet and setup. Sending an Email with Apps Script.GoogleScriptsendemail. Spreadsheet App add content. Test Sheet Google Script. Google Script Trigger Examples. Get Data Parameters. JavaScript Fetch. Outdate HTML in Submit. Google Script Application. Send email on submit. Random ID function and Date Add. Emailtemplate. CreateWebApp.

УЧЕБНО-МЕТОДИЧЕСКАЯКАРТА
(Дневнаяформаобучения)

Номерраздела, темы	Названиераздела, темы	Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	уср	
1	Основы протокола HTTP	2					Дискуссия
2	Понятие и обзор облачных сервисов	2					Дискуссия
3	Обзор языка JS	2					Дискуссия
4	Обзор языка JS		2		2	2	Лаб. работа
5	Обзор языка JS		2		2	2	Лаб. работа
6	ВведениевGoogleAppsScript	2					Дискуссия
7	Интерфейс DocumentApp	2					Дискуссия
8	Интерфейс SpreadsheetApp	2					Дискуссия
9	Интерфейс SpreadsheetApp		2		2	2	Лаб. работа
10	Интерфейс SitesApp	2				*2*	Дискуссия
11	Интерфейс DriveApp	2	2			*2*	Лаб. работа
12	Интерфейс Gmail App	2	2			*2*	Лаб. работа
13	Интерфейс CalendarApp	2			2		Дискуссия
14	Практические исследования		4		2		Лаб. работа
15	Практические исследования		4		2		Лаб. работа
16	Интеграция приложений		4		4	*2*	Лаб. работа
	Всего	20	22		16	14	Зачет

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА
(Заочная форма обучения)

Номерраздела, темы	Названиераздела, темы	Количество аудиторных часов					Количество часов УСР	Форма контроля знаний
		Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	Иное		
1	Основы протокола HTTP							Дискуссия
2	Понятие и обзор облачных сервисов	2						Дискуссия
3	Обзор языка JS							Дискуссия
4	Обзор языка JS							Дискуссия
5	Обзор языка JS							Дискуссия
6	ВведениевGoogleAppsScript	2						Дискуссия
7	Интерфейс DocumentApp							Дискуссия
8	Интерфейс SpreadsheetApp							Дискуссия
9	Интерфейс SpreadsheetApp	2						Дискуссия
10	Интерфейс SitesApp							Дискуссия
11	Интерфейс DriveApp		2					Дискуссия
12	Интерфейс Gmail App		2					Дискуссия
13	Интерфейс CalendarApp							Дискуссия
14	Практические исследования				2			Лаб. работа
15	Практические исследования				2			Лаб. работа
16	Интеграция приложений				2			Лаб. работа
	Всего	6	4		6			Зачет

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Формы контроля знаний

– Дискуссия.
– Лабораторная работа.
– Зачет.

Перечень программного обеспечения

– SublimeText.
– GoogleChrome.

– Node.JS.

Рекомендации по организации и выполнению УСР

Для углубленного самостоятельного изучения учебного материала в рамках УСР выделяются следующие темы дисциплины:

- Основы протокола HTTP
- Базовые типы данных
- Управление вычислительным процессом
- Функции
- Объекты
- Массивы
- Замыкания и область видимости
- Методы объектов и контекст вызова

Самостоятельное изучение материала данных тем преследует цель получения навыков работы с современными версиями программного обеспечения для реализации web-приложений.

Форма выполнения заданий – индивидуальная.

Форма контроля выполнения заданий – дискуссия, реферат, презентация.

ФОРМЫ КОНТРОЛЯ ЗНАНИЙ

1 Отчеты по лабораторным работам.

2 Тестирование.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

ОСНОВНАЯ

1 Флэнаган, Д. JavaScript. Подробное руководство. – М.: Символ-Плюс, 2013. - 1080 с.

2 Резиг, Дж. JavaScript для профессионалов. - СПб.: Питер, 2017. – 240 с.

3 Симпсон, К. ES6 и не только. - СПб.: Питер, 2017. – 336 с.

4 Пауэрс, Ш. Изучаем Node. Переходим на сторону сервера. - СПб.: Питер, 2017. – 304 с.

5 Маклафлин, Б. Объектно-ориентированный анализ и проектирование/ Б. Маклафлин, Г. Поллайс, Д. Уэст – СПб.: Питер, 2013. – 608 с.

6 Защита информации : учебное пособие для студентов вузов по направлению подготовки 210700 "Инфокоммуникационные технологии и системы связи квалификации (бакалавр), (магистр) / А.П. Жук, [и др.], УМО по образованию в области информационных технологий и систем связи. – 2-е изд. – Москва : РИОР : ИНФРА-М, 2015. – 392 с.

ДОПОЛНИТЕЛЬНАЯ

7 Веру, Л. Секреты CSS. Идеальные решения ежедневных задач. - СПб.: Питер, 2016. – 336 с.

8 Браун, И. Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. - СПб.: Питер, 2017. – 380 с.

9 Мельников, В. П. Информационная безопасность : учебник / Владимир Павлович Мельников, Александр Ильич Куприянов, Т.Ю. Васильева, УМО по образованию в области автоматизированного машиностроения ; ред. Владимир Павлович Мельников. – 2-е изд, перераб. и доп. – Москва : КНОРУС, 2018. – 372 с.

10 Информационные технологии. Методы и средства безопасности. Программные средства защиты от воздействия вредоносных программ и антивирусные программные средства. Общие требования : официальное издание : СТБ П 34.101.8-2003: утв.и введен в действ.от 28 апреля 2003 г. 22. – Минск : Госстандарт, 2003.

11 Информационные технологии. Методы и средства безопасности. Профиль защиты программных средств. Система управления сайта : СТБ 34.101.37-2011 : издание официальное. – Минск : Госстандарт, 2012. - 188790. – 287.

12 Вечерко, Е. В. Методы оценки стойкости стеганографической защиты информации на основе марковских случайных процессов : автореферат диссертации ... кандидата физико-математических наук : 05.03.19 / Егор Валентинович Вечерко, Белорусский государственный университет. – Минск : [б.и.], 2016. – 22 с.

ЭЛЕКТРОННЫЕ РЕСУРСЫ

13 Свободная энциклопедия ВикипедиЯ [Электронный ресурс]. – 2019. – Режим доступа: http://ru.wikipedia.org. – Дата доступа: 15.05.2019.

14 Интернет университет информационных технологий [Электронный ресурс]. – 2019. – Режим доступа: http://www.intuit.ru. – Дата доступа: 15.05.2019.

15 Информационно-справочный портал технической информации Хабрахабр [Электронный ресурс]. – 2019. – Режим доступа: http://habr.com. – Дата доступа: 15.05.2019.

16 Информационно-аналитический сайт [Электронный ресурс]. – 2019. – Режим доступа: https://www.ixbt.com. – Дата доступа: 15.05.2019.

17 Google Apps Script Complete Course - Beginner to Advanced [Электронныйресурс]. – 2019. – Режимдоступа: https://www.udemy.com/course/apps-script-course/. – Датадоступа: 15.05.2019.