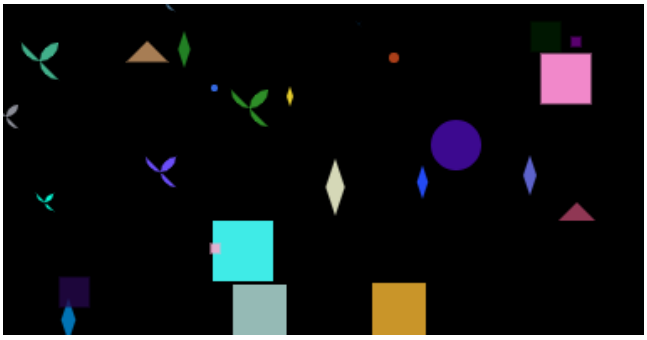
Варианты второго практического задания по курсу «Языки управления приложениями»

При желании можно повысить уровень сложности заданий. Элементы, повышающие уровень сложности, обсуждаются с преподавателем.

Салют (сложность низкая)



Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **салют**.

Правила игры:
На игровом поле в случайных местах появляются разные геометрические фигуры, как будто вспышки салюта. Они постепенно увеличиваются, тускнеют и совсем пропадают. Вид появляющихся фигурок и их количество определяет пользователь до начала игры.

Постановка задачи:

На странице должно быть две области. В одной будет располагаться canvas (игровое поле), во второй - форма, с помощью которой имеется возможность установить начальные настройки и запустить игру. До начала игры пользователь указывает, какие типы фигур могут появляться на игровом поле. Выбор должен быть предоставлен с помощью checkbox из как минимум 6 стандартных типов (круг, квадрат, звезда и т.д.). Также пользователю предоставляется возможность указать интенсивность салюта: большая, средняя, низкая. Когда игра запущена, на игровом поле в случайных местах начинают появляться фигурки выбранных типов. Цвет и размер выбираются случайным образом. Фигурки некоторое время растут, а потом исчезают. Частота появления новых фигур выбирается случайным образом, но зависит интенсивности, выбранной пользователем.

Шары (сложность низкая)

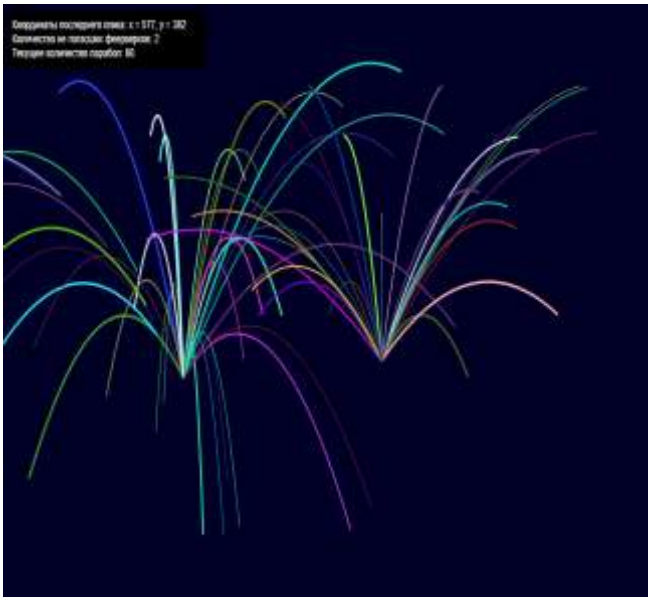
Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **шары**.

Правила игры:
Пользователь нажимает левую кнопку мыши на игровом поле и через какое-то время отпускает. На поле начинает надуваться цветной шар, чем дольше была прижата кнопка мыши, тем больше получается шар.

Постановка задачи:

На странице должно быть две области. В одной будет располагаться canvas (игровое поле), во второй - форма, с помощью которой имеется возможность установить начальные настройки и запустить игру. До начала игры пользователь указывает, какие цвета должны быть у шаров (2-3 фиксированных или случайный). Выбор должен быть предоставлен с помощью checkbox. Также пользователю предоставляется возможность указать скорость надувания шаров: большая, средняя, низкая. Когда игра запущена, на игровом поле в местах клика пользователя мышкой начинают надуваться шары. В центре они почти прозрачные, потом интенсивность цвета нарастает. Размер шара зависит от того, как долго пользователь держал прижатой кнопку мыши. Должна быть предусмотрена возможность очистки игрового поля.

Фейерверк (сложность высокая)

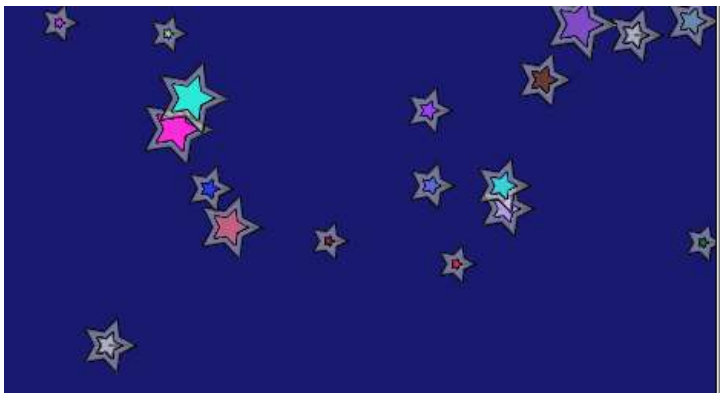


Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **фейервек**.

Правила игры:
Пользователь кликает по игровому полю. От этой точки начинают отходить цветные дуги. По прошествии какого-то времени эти дуги исчезают.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых будет отображаться текущая ситуация игры (координаты последнего клика и количество не погасших фейерверков). Пользователь кликает по игровому полю, от этой точки начинают поточечно отрисовываться параболы таким образом, что сначала движение идет вверх, а потом вниз. Параболы отрисовываются до тех пор, пока не достигнут нижнего, левого или правого края канвы. Параметры параболы и ее цвет определяются случайным образом. Постепенно точки начинают пропадать. Когда все точки от фейерверка пропадают, этот фейерверк удаляется из существующих.

Звездопад (сложность низкая)

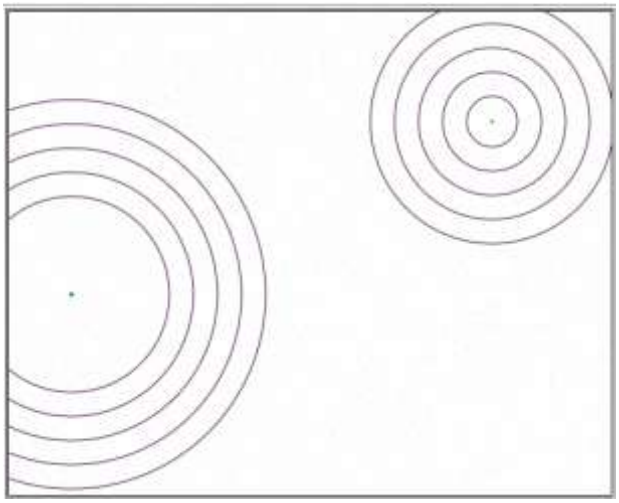


Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **звездопад**.

Правила игры:
Перед началом игры пользователь задает количество звезд на игровом поле. После начала игры звезды начинают появляться сверху игрового поля и лететь вниз под разными углами.

Постановка задачи:
На странице должно быть две области. В одной будет располагаться canvas (игровое поле), во второй - форма, с помощью которой имеется возможность установить начальные настройки и запустить игру. До начала игры пользователь указывает, сколько звезд должно быть на экране одновременно, и запускает игру. Когда игра запущена, на игровом поле начинают появляться звезды. Звезда изображается двумя окружностями (см.вариант молекулы). Размер звезды, ее угол падения и цвет выбираются случайным образом. При исчезновении звезды с экрана сверху тут же появляется новая звезда. Таким образом на игровом поле всегда находится то количество звезд, которое указал пользователь в начале игры. В отдельном текстовом поле в ходе игры показывается количество звезд, которое появилось с момента начала игры.

Круги на воде (сложность средняя)



Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **круги на воде**.

Правила игры:
Пользователь кликает по игровому полю, от этого по игровому полю от этой точки начинают разбегаться круги.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых будет отображаться текущая ситуация игры (количество кругов на поле и общее количество кругов в игре). Пользователь кликает по игровому полю, появляется точка и окружность с центром в этой точки. В следующий момент таких окружностей становится 2, но вторая появляется с большим диаметром, чем первая. И так далее, пока окружностей не станет 5. Как только появляется шестая окружность, самая маленькая из текущих стирается. Так происходит до тех пор, пока все окружности не выйдут за область игрового поля. После этого центр окружности стирается. Кликнув несколько раз по полю пользователь порождает несколько кругов на воде, которые могут пересекаться друг с другом.

Хвостик (сложность высокая)

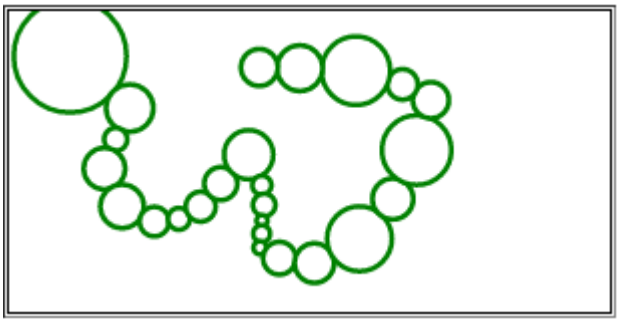
Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **хвостик**.

Правила игры:
Пользователь водит мышкой по игровому полю, за курсором бегает хвостик из точек. Если мышка останавливается, точки собираются в кучку. При возобновлении движения мыши, точки из кучки опять растягиваются в хвостик.

Постановка задачи:
На странице должно быть две области. В одной будет располагаться canvas (игровое поле), во второй - форма, с помощью которой имеется возможность установить начальные настройки и запустить игру. До начала игры пользователь указывает сколько точек он хочет видеть в хвостике, их размер и цвет. Когда игра запущена, на игровом поле за мышкой начинают следовать несколько точек. Расстояние между точками зависит от скорости движения мышкой. При остановке курсора все точки собираются в кучу около него, т.е. рисуются в одном и том же месте. Как только курсор продолжает движение, точки постепенно вылезают из кучи и следуют за курсором, выстроившись в хвостик.

Пузыри (сложность средняя)

Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **пузыри**.

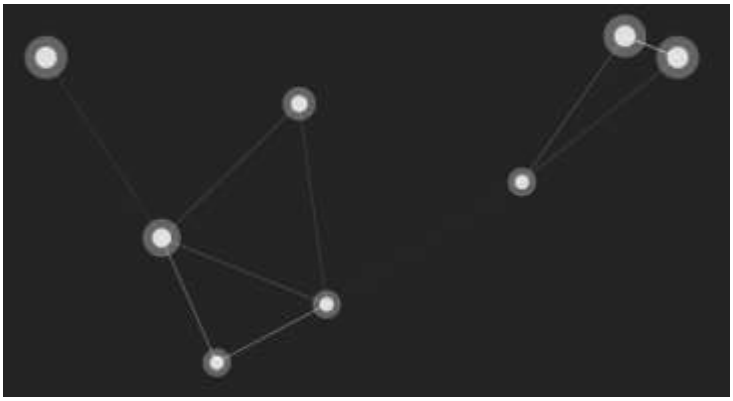


Правила игры:
Пользователь кликает на игровом поле. Каждый клик надувает новый пузырь. Все пузыри касаются друг друга, таким образом получается дорожка из пузырей.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых будет отображаться текущая ситуация игры (радиус и центр текущей окружности). Пользователь кликает по игровому полю, появляется новая окружность, касающаяся предыдущей. При этом центр новой окружности лежит на отрезке от центра предыдущей окружности до места клика, а радиус таков, что в месте клика новая окружность заканчивается (см.рисунок).
Этот вариант можно **усложнить**: пузыри строятся не по клику, а следуя за движением мышки. Чем быстрее мышка двигается, тем больше пузыри получаются. Старые пузыри нужно при этом удалять.

Молекулы (сложность средняя)

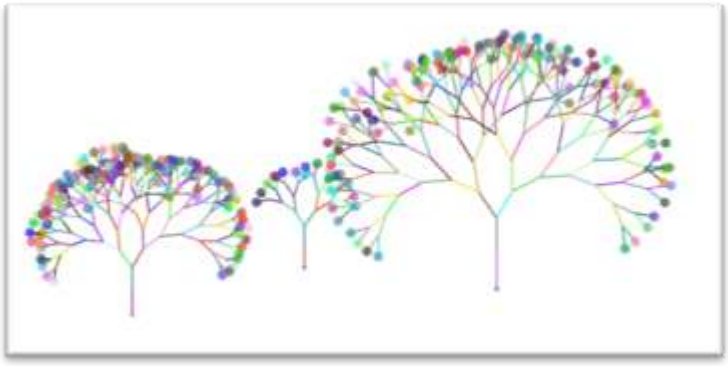
Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **молекулы**.



Правила игры:
Пользователь кликает на игровом поле. Каждый клик порождает новую молекулу. Новая молекула соединяется связями с уже существующими.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть расположено несколько текстовых полей для отображения текущей ситуации игры (координаты последней молекулы, их общее количество). После каждого клика по игровому полю рисуется молекула. Молекулы бывают трех типов: большие, средние и малые. На игровом поле молекула это совокупность двух окружностей с центром в точке клика (маленькая белая, большая серая), радиус этих окружностей зависит от размера молекулы. Дальше высчитывается расстояние от этой новой молекулы до остальных. В зависимости от этого расстояния и размера соединяемых молекул меняется толщина отрезка, соединяющего центры молекул: большие молекулы притягивают к себе другие с большей силой, чем маленькие; чем ближе молекулы друг к другу, тем сильнее связь. Для отдаленных молекул толщина такого отрезка равна нулю, т.е. ее можно не рисовать.
Этот вариант можно **усложнить**: молекулы двигаются каждая со своей скоростью, соответственно и отрезки, соединяющие их, то появляются и утолщаются, то становятся тоньше и пропадают.

Дерево (сложность средняя)



Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **дерево**.

Правила игры:
Пользователь кликает на игровом поле и от этой точки начинает расти дерево, его размер и количество ветвей задаются до начала игры.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых имеется возможность установить начальные настройки. До начала игры пользователь указывает, сколько веток должно отходить от каждой развилки (от 2 до 4), их длина (2-3 фиксированных значения или случайная для каждой ветки, но в рамках фиксированных границ; по мере роста дерева длина сегментов автоматически уменьшается по сравнению с предыдущими), какое количество сегментов «вглубь» нужно нарисовать, цвет сегментов (2-3 фиксированных или случайный), нужны ли листочки на концевых сегментах. После клика дерево отрисовывается в соответствии с указанными параметрами. Отрисовка должна быть плавной по уровням (сначала первый уровень от корня, потом второй и т.д.). В качестве листочков можно брать окружности или рисовать ромбы. Когда дерево полностью нарисовано, пользователь может изменить параметры и начать рисовать еще одно, не стирая предыдущего. Должна быть предусмотрена возможность очистки игрового поля для начала рисования нового леса.
Вариация: можно взять любой несложный фрактал и реализовывать его по тем же принципам.

Облака (сложность низкая)

Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **облака**.

Правила игры:
Пользователь кликает на игровом поле и от этой точки начинает расти облако, его размер и количество окружностей задаются до начала игры.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых имеется возможность установить начальные настройки. Пользователь может указать максимальное и минимальное количество окружностей у облака, его цвет (несколько на выбор или случайный оттенок серого) и размер (большое, среднее или маленькое – от этого будет зависеть радиус окружностей и удаленность центров окружностей друг от друга). После клика в соответствии с указанными параметрами случайным образом определяется цвет, количество окружностей и размер каждой из них, после этого облако отображается на игровом поле. Когда облако полностью нарисовано, пользователь может изменить параметры и начать рисовать еще одно, не стирая предыдущего. Должна быть предусмотрена возможность очистки игрового поля для начала рисования нового неба.
Этот вариант можно **усложнить**: облака двигаются по небу, каждое со своей скоростью, имитируя ветер. Нижние облака двигаются немного быстрее верхних. В настройках можно задать направление ветра или остановить его.

Аэрохоккей (сложность высокая)

Цель: клиентское приложение на языке JavaScript, реализующее интерфейс игры **аэрохоккей**.

Правила игры:
Пользователь двигает мышкой по игровому полю, на котором нарисованы одни ворота и шайба. Если мышка касается шайбы, то шайба начинает двигаться в том направлении и с такой скоростью, с которой двигался курсор в момент касания. Если шайба касается ворот, то засчитывается гол и шайба выставляется на центр поля.

Постановка задачи:
На странице выделяется игровое поле (canvas). Рядом или ниже должно быть несколько текстовых полей, в которых имеется возможность установить начальные настройки. До начала игры пользователь указывает силу трения (высокая, средняя или низкая – от этого будет зависеть скорость замедления шайбы до полной остановки) и размер шайбы. Так же должно быть поле для отображения текущего счета игры. После начала игры нужно следить, как быстро передвигается курсор и не коснулся ли он шайбы (центр окружности и радиус мы всегда знаем), если касание произошло, то шайба начинает движение с той же скоростью и в том же направлении, что двигался курсор в момент касания, скорость движения шайбы постепенно замедляется. При ударе о край поля шайба отражается от него и продолжает движение. Игра останавливается после 5 голов, пользователю показывается время, которое он потратил на игру.

При желании можно выбрать свою игру, но выбор ОБЯЗАТЕЛЬНО обсудить с преподавателем.