JavaScript Turtle Graphics Library

Середовище для програмування Черепашачої графіки

Вступ

`JavaScript Turtle Graphics Library` - це навчальне середовище для програмування, яке містить базовий набір інструментів для створення Черепашачої графіки (`Turtle Graphics`), використовуючи об'єктно-орієнтований підхід.

Мета створення середовища `JavaScript Turtle Graphics Library` - знайомство з основами програмування на прикладі роботи з Черепашачою графікою у вебпереглядачі, використовуючи мову програмування `JavaScript` і можливості бібліотеки p5.js.

Підключення

Щоб застосувати бібліотеку `JavaScript Turtle Graphics Library` у своєму проєкті, її необхідно підключити, використовуючи один із способів:

1. В онлайн-редакторі editor.p5js.org файл бібліотеки `JavaScript Turtle Graphics Library` необхідно підключити у файлі `index.html` перед файлом ескізу `sketch.js` .
2. У разі локальної розробки завантажити зразок проєкту `p5.js` , у який підключити файл бібліотеки `JavaScript Turtle Graphics Library` у файлі `index.html` перед файлом ескізу `sketch.js` .
3. У онлайн-редакторі CodePen: -- перейти у вкладку `JS` ; -- натиснути на зображення шестерні; -- у розділі `Add External Scripts/Pens` у рядку пошуку знайти і обрати `p5.js` ; -- у розділі `Add External Scripts/Pens` натиснути на кнопку `+add another resource` і у рядок, що з'явився, додати посилання на файл бібліотеки `JavaScript Turtle Graphics Library` (посилання з розділу Початковий код); -- зберегти зміни, натиснувши кнопку `Save & Close` .
4. В онлайн-середовищі Replit при створенні нового `Repl` необхідно обрати шаблон `p5.js` , відвантажити файл бібліотеки `JavaScript Turtle Graphics Library` у створений `Repl` і підключити файл бібліотеки у файлі `index.html` перед файлом ескізу `script.js` .
5. У онлайн-середовищі OpenProcessing у створений ескіз за допомогою розділу `FILES` відвантажити файл бібліотеки `JavaScript Turtle Graphics Library` . Перейшовши у розділ `SKETCH` і обравши режим (`MODE`) `HTML/CSS/JS` , підключити відвантажений файл бібліотеки у файлі `index.html` перед файлом ескізу `mySketch.js` .
6. Використати p5js-widget.pp.ua - простий онлайн-редактор, створений на основі p5.js-widget - інструменту для вбудовування на сторінки сайтів редактора для запуску і редагування ескізів `p5.js` . У цей редактор бібліотека `JavaScript Turtle Graphics Library` вже інтегрована і готова до використання.

> *Завантажити файл бібліотеки `JavaScript Turtle Graphics Library` можна у розділі Початковий код.*

Інтерактивна мапа

Для зручності роботи із `JavaScript Turtle Graphics Library` створений інтерактивний застосунок, за допомогою якого можна:

- відслідковувати, фіксувати на полотні і, за потреби, зберігати координати у текстовий файл;
- зберігати у файл полотно з малюнком;
- змінювати властивості Черепашки/олівця (колір, форму, розмір, кутову орієнтацію Черепашки, товщину і колір олівця).

Спробувати `JavaScript Turtle Graphics Library` та інтерактивний застосунок в дії можна за покликаннями:

- JavaScript Turtle Graphics Library
- Інтерактивна мапа

> *За потреби, перейшовши за цими покликаннями, можна зробити `FORK`-и ескізів.*

Огляд можливостей

Початкові налаштування

У блоці `setup()` записуємо функцію `createCanvas(windowWidth, windowHeight)` для створення полотна, де `windowWidth` і `windowHeight` - це системні змінні, що містять значення ширина і висота внутрішнього вікна полотна відповідно, і викликаємо `myCode()` - це назва блоку, в якому записані інструкції для Черепашки.

```
let t;

function setup() {
  createCanvas(windowWidth, windowHeight);
  myCode();
}
```

> *Значення розмірів полотна, які за замовчуванням використовує функція `createCanvas()` (коли викликається без аргументів) - `200x200` пікселів. За потреби можна створити полотно будь-якого розміру, зазначивши у `createCanvas()` відповідні значення розмірів ширини і висоти.*

Також на початку у коді оголосили ім'я для майбутньої Черепашки - `t`.

> *Ім'я для Черепашки можна обрати на свій задум.*

Блок `myCode()` має такий вигляд:

```
function myCode(){
  // інструкції
}
```

> *Усі інструкції для керування Черепашкою записуємо у блоці `myCode()`. Назву цього блоку можна змінити на свій задум.*

Завжди першою інструкцією у блоці `myCode()` є інструкція зі створення об'єкта Черепашки з певним ім'ям, оголошеним раніше:

```
function myCode(){
  t = new Turtle();
}
```

Тепер Черепашка отримала своє ім'я `t`. Усі наступні інструкції необхідно записувати у форматі `t.інструкція`.

Координатна сітка

Якщо необхідно створити координатну сітку на полотні, додаємо у `myCode()` інструкцію `grid()`:

```
function myCode(){
  t = new Turtle();
  t.grid();
}
```

Інструкція `grid()` використовує три параметри із такими значеннями за замовчуванням:

- крок сітки - `50` пікселів,
- колір сітки - Platinum,
- колір осей - Cerulean.

> *Значення кольору записується як рядок в одному із форматів: `"red"` (назва), `"#fdfd90"` (шістнадцяткове значення), `"rgb(11, 156, 78)"` (значення червоної, зеленої, синьої складових), `"rgba(45, 145, 67, 0.5)"` (значення червоної, зеленої, синьої складових і прозорості).*

Значення за замовчуванням використовуються тоді, коли `grid()` викликається без аргументів, як у прикладі вище. За потреби `grid()` можна викликати із користувацькими значеннями, зазначивши їх у дужках у вказаному порядку.

> *Використовуючи інструкцію `setStepGrid(step)` перед малюванням сітки можна окремо встановити крок сітки `step`, а за допомогою інструкції `getStepGrid()` можна отримати поточне значення кроку сітки.*

Черепашка і `p5.js`

На полотні поруч з Черепашкою можна створювати зображення, використовуючи інструменти бібліотеки `p5.js`. Виклики функцій `p5.js` для цих цілей записуються, як відомо, у тілі функцій `draw()` чи `setup()`. Зверніть увагу, що за замовчуванням початок координат `(0, 0)` знаходиться у лівому верхньому куті полотна.

Початок координат `(0, 0)` для Черепашки міститься в центрі полотна. За таких умов, одночасно відслідковувати координати для Черепашки і для побудови фігур складно.

У цьому разі код для побудови фігур за допомогою інструментів бібліотеки `p5.js` можна записувати у блоці `myDraw()` (за потреби назву блоку можна змінити на іншу) між коментарями `// початок коду` і `// кінець коду`, а виклик `myDraw()` помістити, наприклад, у блоці `draw()`, як показано нижче:

```
function draw() {
  myDraw();
}

function myDraw(){
  push();
  translate(width / 2, height / 2);
  scale(1, -1);
  // початок коду
```

```
  let [x, y] = t.getPosition();
  console.log(x, y);

  // кінець коду
  pop();
}
```

> *За допомогою виразу `let [x, y] = t.getPosition();` можна отримати поточні координати Черепашки з ім'ям `t` в координатній сітці, початок координат якої міститься в центрі полотна, а далі отримані координати використати для побудови фігур за допомогою інструментів бібліотеки `p5.js` на полотні.*

Відображення Черепашки, інформаційна панель та компас

За замовчуванням на полотні відображається інформаційна панель з даними про:

- стан Черепашки (кутову орієнтацію, поточні координати),
- стан олівця (на полотні чи піднятий),
- координати вказівника миші.

а у правому нижньому куті полотна увімкнений компас, який вказує напрямок і значення кутової орієнтації Черепашки.

За відображення вищезгаданих елементів інтерфейсу відповідають інструкції, які записуються у блоці `draw()`

```
function draw() {
  t.place();
  t.compass();
  t.dashboard();
}
```

і використовуються для таких цілей:

- `place()` - показати Черепашку на полотні у її поточних координатах. Цю інструкцію рекомендується завжди використовувати.
- `compass()` - показати компас.
- `dashboard()` - показати інформаційну панель.

> *Також на екрані можна відобразити ім'я розробника: у блоці `draw()` розмістити інструкцію `t.creator()`.*

Багато Черепашок

Для створення двох (або більше) Черепашок, необхідно оголосити їхні імена та створити їх із цими іменами:

```
let t1, t2;

function setup() {
  createCanvas(windowWidth, windowHeight);
  myCode();
}
```

```
function myCode(){
  t1 = new Turtle();
  t2 = new Turtle();
}
```

Відображення на полотні створених Черепашок відбувається за допомогою виклику інструкцій `place()` для кожної із них у блоці `draw()`:

```
function draw() {
  t1.place();
  t2.place();
}
```

Відповідно інструкції для різних Черепашок записуються у форматі `t1.інструкція`, `t2.інструкція` і т. д.

> *У режимі кількох Черепашок інформаційна панель і компас показують дані для одного екземпляра Черепашки, для якого вони викликаються.*

Зміна розмірів вікна полотна/вебпереглядача

При зміні розмірів вікна вебпереглядача/полотна усі елементи інтерфейсу (мапа, інформаційна панель, компас тощо) залишаються на своїх місцях і з'являються смуги прокручування. Щоб ці елементи налаштувалася відповідно до нових розмірів, оновіть вебсторінку. За потреби перед цим збережіть свій код.

Якщо ви працюєте на вебсторінці інтерактивної мапи, спочатку встановіть розміри вікна вебпереглядача, а потім оновіть вебсторінку, щоб елементи інтерфейсу налаштувались відповідно до цих розмірів.

Інструкції

Рух

> *Рух Черепашки відбувається **без анімації**, тобто виконуються усі записані інструкції і Черепашка відразу розташовується на полотні у точці з кінцевими координатами.*

Після запуску Черепашка з'являється на полотні:

- у точці з координатами `(0, 0)` (центр полотна),
- "дивиться" праворуч,
- має початкову кутову орієнтацію, що вимірюється у градусах, `0°`.

Інструкція	Опис
`forward(distance)`	Перемістити Черепашку вперед на відстань `distance`, у напрямку, куди "дивиться" Черепашка.
`back(distance)`	Перемістити Черепашку назад на відстань `distance`, у протилежний бік напрямку її руху.
`goto(x, y)`	Перемістити Черепашку в точку з координатами `(x, y)`.
`home()`	Перемістити Черепашку в точку з координатами `(0, 0)` (центр полотна) і встановити початкову кутову орієнтацію.
`left(angle)`	Повернути Черепашку ліворуч на кут `angle`, де `angle` - число (ціле

	чи з плаваючою крапкою). Якщо значення кута `angle` додатне, то Черепашка повертається на це значення кута проти годинникової стрілки (кутова орієнтація стає `angle`), якщо від'ємне - Черепашка повертається за годинниковою стрілкою на це значення кута (кутова орієнтація стає `360 - abs(angle)`).
`right(angle)`	Повернути Черепашку праворуч на кут `angle`, де `angle` - число (ціле чи з плаваючою крапкою). Якщо значення кута `angle` від'ємне, то Черепашка повертається на це значення кута проти годинникової стрілки (кутова орієнтація стає `angle`), якщо додатне - Черепашка повертається за годинниковою стрілкою на це значення кута (кутова орієнтація стає `360 - abs(angle)`).
`setHeading(angle)`	Встановити кутову орієнтацію Черепашки на кут `angle`, де `angle` - число (ціле чи з плаваючою крапкою). Якщо значення кута `angle` додатне, то Черепашка повертається на це значення кута проти годинникової стрілки (кутова орієнтація стає `angle`), якщо від'ємне - Черепашка повертається за годинниковою стрілкою на це значення кута (кутова орієнтація стає `360 - abs(angle)`).

Черепашка

Інструкція	Опис
`getPosition()`	Отримати поточні координати Черепашки у вигляді списку `[x, y]`.
`getHeading()`	Отримати поточну кутову орієнтацію Черепашки.
`setShape(shape)`	Встановити форму `shape` для Черепашки. `shape` може набувати значень: `"blank"` (невидимість), `"circle"`, `"square"`, `"triangle"`. За замовчуванням форма Черепашки `"triangle"`.
`setShapeSize(s)`	Встановити розмір `s` для форми Черепашки. За замовчуванням - мінімальний розмір `1`, максимальне значення `20`.
`showTurtle()`	Зробити Черепашку видимою.
`hideTurtle()`	Зробити Черепашку невидимою.
`clr()`	Очистити полотно. Черепашка залишається у поточній точці, її кутова орієнтація зберігається.

Олівець

> *Якщо олівець на полотні, при переміщенні Черепашка малюватиме лінію.*

Інструкція	Опис
`penUp()`	Підняти олівець.
`penDown()`	Опустити олівець на полотно. За замовчуванням - олівець на полотні.
`setPenSize(s)`	Встановити у пікселях товщину лінії олівця на `s`. За замовчуванням товщина лінії олівця `1` піксель.

Колір

> *Значення кольору записується як рядок в одному із форматів: `"red"` (назва), `"#fdfd90"` (шістнадцяткове значення), `"rgb(11, 156, 78)"` (значення червоної, зеленої, синьої складових), `"rgba(45, 145, 67, 0.5)"` (значення червоної, зеленої, синьої складових і прозорості).*

Інструкція	Опис
`setColor(c, f)`	Встановити колір `c` для олівця та колір заливки `f` для зафарбовування фігур. За замовчуванням колір олівця - чорний, колір заливки - прозорий. Для встановлення лише кольору олівця інструкція використовується з одним параметром: `setColor(c)` .
`setBgColor(c)`	Встановити колір `c` для тла полотна. За замовчуванням - білий.
`setFillColor(f)`	Встановити колір заливки `c` для зафарбовування фігур. За замовчуванням - заливка прозора.
`beginFill()`	Увімкнути зафарбовування фігури поточним кольором заливки.
`endFill()`	Вимкнути зафарбовування фігури поточним кольором заливки.

Фігури

Інструкція	Опис
`oval(r, e)`	Намалювати коло чи еліпс. Якщо параметри `r` і `e` набувать однакових значень, то Черепашка малює коло зі значенням радіуса, що дорівнює `r`. У разі різних значень `r` і `e` - отримуємо еліпс.
`polygon([[x1, y1], [x2, y2], [x3, y3], ...])`	Намалювати багатокутник за координатами вершин `x1, y1, x2, y2, x3, y3, ...` (за годинниковою стрілкою).

Текст

Інструкція	Опис
`write("txt", {horizontal: A, vertical: B}, {font: C, size: D, style: E})`	Написати текст `"txt"` у поточній позиції Черепашки. Об'єкт `{horizontal: A, vertical: B}` використовується для вирівнювання тексту, де `A` може набувати значень `LEFT`, `CENTER` або `RIGHT`, а `B` - `TOP`, `BOTTOM`, `CENTER` або `BASELINE`. Об'єкт `{font: C, size: D, style: E}` визначає шрифт, розмір тексту і стиль відповідно, а саме: `C` може набувати значень `Arial`, `Times`, `Verdana` тощо, `D` - ціле число, `E` може набувати значень `NORMAL`, `ITALIC`, `BOLD`, `BOLDITALIC`.

> *За замовчуванням текст є порожнім рядком і має такі параметри: вирівнювання `{horizontal: CENTER, vertical: CENTER}` , шрифту `{font: "sans-serif", size: 12, style: NORMAL}` .*

Розробка

Для розробки середовища `JavaScript Turtle Graphics Library` застосовувалась мова програмування `JavaScript` та інструменти бібліотеки [p5.js](p5.js).

Початковий код

Початковий код бібліотеки `JavaScript Turtle Graphics Library` зберігається в єдиному файлі, який можна завантажити тут.

Ліцензія

Використання бібліотеки `JavaScript Turtle Graphics Library` визначається умовами ліцензії `GNU General Public License` (`GPL v3`).

Код бібліотеки можна вільно і безкоштовно копіювати, змінювати на свій задум та розповсюджувати.
