

JavaScript Turtle Graphics

Середовище для програмування Черепащої графіки

Вступ

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

Мета створення середовища JavaScript Turtle Graphics - знайомство з основами програмування на прикладі роботи з Черепащою графікою у вебпереглядачі, використовуючи мову програмування JavaScript і можливості бібліотеки [p5.js](#).

Підключення

Середовище JavaScript Turtle Graphics можна вважати бібліотекою, яка розширює [p5.js](#).

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

1. В онлайн-редакторі [editor.p5js.org](#) файл бібліотеки JavaScript Turtle Graphics 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 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 (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 у створений Repl і підключити файл бібліотеки у файлі index.html перед файлом ескізу script.js .
5. У онлайн-середовищі [OpenProcessing](#) до створеного ескізу у розділі FILES відвантажити у ескіз файл бібліотеки JavaScript Turtle Graphics . 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 вже інтегрована.

Завантажити файл бібліотеки JavaScript Turtle Graphics можна у розділі [Початковий код](#).

Інтерактивна мапа

Середовище JavaScript Turtle Graphics доповнюється інтерактивним застосунком з назвою Інтерактивна мапа , який може:

- 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 середовище `JavaScript Turtle Graphics` та інтерактивний застосунок в дії можна за покликаннями:

- [JavaScript Turtle Graphics](#)
- [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 (ім'я обрали вище):

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

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

Мапа

Якщо необхідно створити мапу (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 засоби бібліотеки `p5.js`. У цьому разі код записуємо у блоці `myDraw()` (за потреби назву блоку можна змінити на іншу) між коментарями `// початок коду` і `// кінець коду`. Також назву блоку `myDraw()` необхідно записати у блоці `draw()`, як показано нижче:

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

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

  // кінець коду
  pop();
}
```

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()`.

Зміна розмірів вікна вебпереглядача/полотна

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 налаштувалися на нові розміри, оновіть вебсторінку. За потреби перед цим збережіть свій код.

Якщо ви працюєте на вебсторінці інтерактивної мапи (p5js-widget.pp.ua/map/), спочатку встановіть розміри вікна вебпереглядача, а потім оновіть вебсторінку, щоб елементи інтерфейсу налаштувались під ці розміри.

Інструкції для Черепашки, згруповані за категоріями

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
<code>forward(distance)</code>	Перемістити Черепашку вперед на відстань <code>distance</code> , у напрямку, куди "дивиться" Черепашка.
<code>back(distance)</code>	Перемістити Черепашку назад на відстань <code>distance</code> , 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 $360 - \text{abs}(\text{angle})$).
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 $360 - \text{abs}(\text{angle})$).
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 $360 - \text{abs}(\text{angle})$).

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 набувають 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 застосовувалась мова програмування JavaScript та інструменти бібліотеки [p5.js](#).

Початковий код

Початковий код JavaScript Turtle Graphics зберігається в єдиному файлі, який можна завантажити [тут](#).

Ліцензія

Середовище JavaScript Turtle Graphics поширюється вільно і безкоштовно.

© 2023. JavaScript Turtle Graphics . Розроблено з  Олександр Мізюк