Tricky Turkeys: JavaScript Teacher Guide

Summary

- Coding skill level: **Advanced**
- Recommended grade level: **Grades 6+ (U.S.), Years 7+ (U.K)**
- Time required: **40 minutes**
- Number of modules: **1 module**
- Coding Language: **JavaScript**

Teacher Guide Outline

Welcome!

Let's find some turkeys! Students will begin today's coding adventure by completing the "Getting Started" activity, which instructs them to search for fall-themed images and save them. Next, they'll move on to the step-by-step tutorial in the Text Code Editor where they will use JavaScript and the HTML canvas to create a Tricky Turkeys game. *How to play:* Turkeys are hiding behind obstacles, and the player needs to click on the obstacles to see if a turkey is hiding behind it! Students are provided code to get started, but are encouraged to modify their code to make it their own. The tutorial also includes a Bonus Challenges section, which provides students with ideas on how to enhance their game. **Note:** Students only need to edit the **main.js** file.

How to Prepare

This activity is designed for self-directed learning. Your role will be to help students individually and facilitate as students complete the coding activities on their own. The best way to prepare is to:

1. **Familiarize yourself with the material.** After selecting your Tynker lesson (e.g., Tricky Turkeys), read through this teacher guide and complete the activity before assigning it to students. This will allow you to troubleshoot anything in advance and plan for potential questions from your students.
2. **OPTIONAL: Sign up for a teacher account.** Although an account is NOT required, creating a free teacher account will allow you to access teacher guides, answer keys, and tons of additional resources. You'll also be able to create free accounts for your students, monitor their progress, and see their projects.
3. **OPTIONAL: Create student accounts.** From your teacher account, you can easily create free student accounts for all your students. This will allow them to save their projects and progress, so they can continue coding when they get home! Again, this is not necessary to complete the Tricky Turkeys lesson.

Activity

Overview

Objectives

Students will...

- Apply coding concepts to create a Tricky Turkeys game using JavaScript

Materials

- Computers, laptops, or Chromebooks (1 per student)

Vocabulary

- **Code:** The language that tells a computer what to do
- **Sequence:** The order in which steps or events happen
- **Function:** A set of known actions that the computer can perform

- **Variable:** Stores a value at a named location such as a number or a string of text
- **Argument:** Value passed into a function
- **Parameter:** Variable that holds the value passed as an argument into the function
- **HTML Canvas:** The HTML element that is used to draw graphics
- **Loop:** An action that repeats one or more commands over and over

U.S. Standards

- **CCSS-ELA:** RI.6.4, RI.6.7, SL.6.1, RI.7.4, SL.7.1, SL.8.1, RI.8.4, RI.9-10.5
- **CCSS-Math:** MP.1
- **CSTA:** 2-AP-13, 2-AP-17, 3A-AP-17, 3B-AP-11
- **CS CA:** 6-8.AP.13, 6-8.AP.16, 6-8.AP.17, 9-12.AP.12, 9-12.AP.16
- **ISTE**: 1.c, 1.d, 4.d, 5.c, 5.d, 6.b

U.K. Standards

National Curriculum in England (computing):

- **Key Stage 3 (Years 7-9)**
  - Create, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability
  - Understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct, and know how to report concerns
- **Key Stage 4 (Year 10)**
  - Develop their capability, creativity and knowledge in computer science, digital media and information technology
  - Develop and apply their analytic, problem-solving, design, and computational thinking skills
  - Understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to report a range of concerns

Getting Started (10 minutes)

Note: This activity encourages students to use online resources to find images for their project.

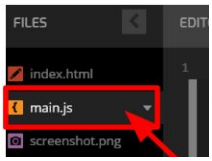
- Inform students that they're going to program a game using JavaScript where turkeys are hiding behind obstacles. Prepare students for today's coding activity by instructing them to use an online source of their choice or Tynker's Media Library and search for 3-5 images that the turkeys can hide behind (e.g., corn, hat, leaf).

Tutorial (30 minutes)

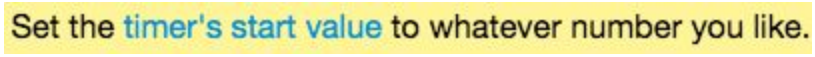
This lesson has one tutorial. Facilitate as students complete the Tricky Turkeys tutorial on their own:

Tricky Turkeys (Tutorial)

- In this DIY, students will follow a step-by-step tutorial in the Text Code Editor, where they will create a game using JavaScript where turkeys are hiding behind obstacles! *How to play:* The player will need to click on the different obstacles to find the hiding turkeys before time runs out.
- Coding activities include using a timer and an event listener, adding images, using a grid (2D array) to set up the turkeys/obstacles, and making your game interactive by adding code to the **clickHandler()** function.
- Make sure students select the **main.js** file (located on the left side of their screen) before they start coding their project. Also point out to students that they only need to edit the **main.js** file:



- Point out to students that they can click the blue hyperlinks in the tutorial, which will take them to specific lines in their code. Here's a screenshot example:



- Are students struggling to upload pictures (“Step 3” of the tutorial)? Check that they have pictures downloaded onto their device. Next, check that they’re reading the provided instructions carefully.
- When students get to "Steps 6-7" of the tutorial, point out that their turkey grid and obstacle grid need to be the same size (e.g., 3x3).
- Do students need help running their program?
    - Students can run their code’s output by selecting this Play button, located at the top right corner of their screen:

- Are students struggling with their code?
    - Check that they’re carefully reading the provided tutorial instructions, then ask them to compare their code with the provided “Your code should look similar to this” sections.
- Did students finish early? Direct their attention to the Bonus Challenges section in the tutorial. Here are some hints to help them get started:
    - **Adjust the scale of the turkey and obstacles-** Refer students to the **turkeyScale** and **obstacleScales** variables, then encourage them to explore different values:

```
var turkeyFound = [0, 0, 0];
var turkeyGridX = [0, 0, 0];
var turkeyGridY = [0, 0, 0];
var turkeyScale = 0.1;

var obstacles = [imgPumpkin];
var obstacleScales = [0.1];
```

Note: A smaller number will decrease the size of the image, whereas a larger number will increase the size of the image.

- **Change the values of the gridScale and topMargin variables-** Encourage students to experiment with positive values. Here's an example:

```
var gridScale = 275;
var topMargin = 20;
```

- **Edit the background color-** Refer students to this part of their code:

```
function draw() {
  // Clear screen
  ctx.fillStyle = "orange";
  ctx.fillRect(0, 0, canvas.width, canvas.height);
```

Next, ask them to modify the variable named **ctx.fillStyle**, and experiment with different colors such as teal, darkgreen, skyblue, yellow, and more! Here's an example:

```
ctx.fillStyle = "skyblue";
```

- **Edit the timer message font-** Refer students to this part of their code:

```
// Draw timer text
ctx.font = "24px Helvetica";
ctx.textAlign = "center";
ctx.fillStyle = "black";
ctx.fillText(timerMessage, canvas.width * 0.5, 30);
```

Next, ask them to modify the variable named **ctx.font**, and experiment with different pixel values (e.g., 30px) and font-families such as cursive, courier, and more! Here's an example:

```
ctx.font = "30px cursive";
```

- **Change the timer message-** Refer students to these two different sections in their code:

1:

```
// Determine timer text.
if (turkeysLeft == 0 && timer > 0) {
  timerMessage = "Congratulations!";
} else if (timer > 0) {
  timerMessage = "You have " + timer + " seconds to find " + turkeysLeft + " turkeys!";
} else {
  timerMessage = "Oops, time's up! Try again."
}
```

2:

```
timerMessage = "You have " + timer + " seconds to find " + turkeyFound.length + " turkeys!";
draw();
setInterval(updateTimer, 1000);
```

Next, ask them to modify the variable named **timerMessage**. Their new message might look like this:

```
timerMessage = turkeyFound.length + " turkeys are hiding! " + "Find them within " + timer + " seconds!";
```

Important: Make sure students change the message in both sections.

Extended Activities

Unplugged Activity: I'm Thankful For...

In this activity, students will reflect on how our lives are affected by STEM (science, technology, engineering, math). Group students into teams of 4 and ask them to discuss the following:

- What scientific breakthroughs are you thankful for? What do you think life would be like if these scientific breakthroughs didn't occur?
- What technological artifact are you most thankful for? Why?
- What product makes you thankful that an engineer improved it? Discuss how the product was made better.
- What are 2-3 different activities that you enjoy doing, which makes you thankful for math (e.g., playing sports/music, cooking)?

Going Beyond Tricky Turkeys

If your students enjoyed Tricky Turkeys, they're sure to enjoy the rest of what Tynker has to offer! Tynker offers a complete premium solution for schools to teach computer science. Over 400 hours of lessons are available to take K-8 students from block coding to advanced text coding. We offer tons of resources for teachers, including comprehensive guides, free webinars, and a forum to connect with other educators.

Do More with Tynker

With Tynker, kids don't just acquire programming skills—they explore the world of possibilities that coding opens up. Tynker has several interest-driven learning paths that make coding fun, both inside and outside the classroom:

- **Coding and Game Design:** Your students can use Tynker Workshop, a powerful tool for crafting original programs to make games, stories, animations, and other projects. They can even share their work with other kids in the Tynker Community.
- **Drones and Robotics:** Tynker integrates with connected toys, including Parrot drones and Lego WeDo robotics kits, so kids can see their code come to life.

- **Minecraft:** Tynker integrates with Minecraft so your students can learn coding through a game they love. Tynker offers skin and texture editing, as well as a custom Mod Workshop that lets kids try their original code in Minecraft.

Tynker for Schools

Used in over 90,000 schools, our award-winning platform has flexible plans to meet your classroom, school, or district needs. All solutions include:

- Grade-specific courses that teach visual coding, JavaScript, Python, robotics and drones
- A library of NGSS and Common Core compliant STEM courses that are great for project-based learning
- Automatic assessment and mastery charts for whole schools and individual classes and students
- Easy classroom management with Google Classroom and Clever integration
- Professional training, free webinars and other teacher training resources

Need help getting Tynker started at your school? Contact us to learn more about teaching programming at your school with Tynker!

Help

Need help? Below you'll find answers to frequently asked questions about using Tricky Turkeys.

How do I prepare for Tricky Turkeys?

1. **Familiarize yourself with the material.** After selecting your Tynker lesson (e.g., Tricky Turkeys), read through this teacher guide and complete the activity before assigning it to students. This will allow you to troubleshoot anything in advance and plan for potential questions from your students.
2. **OPTIONAL: Sign up for a teacher account.** Although an account is NOT required, creating a free teacher account will allow you to access teacher guides, answer keys, and tons of additional resources. You'll also be able to create free accounts for your students, monitor their progress, and see their projects.
3. **OPTIONAL: Create student accounts.** From your teacher account, you can easily create free student accounts for all your students. This will allow them to save their projects and progress, so they can continue coding when they get home! Again, this is not necessary to complete the Tricky Turkeys lesson.

Who is this activity for?

Tricky Turkeys is intended for students in grades 6+ (U.S.) or years 7+ (U.K.) with some coding experience.

Do I need to create Tynker accounts for my students?

No, you do not need to create Tynker accounts for your students.

What devices do I need?

Computers, laptops, or Chromebooks (1 per student) with an internet connection

What will my students learn?

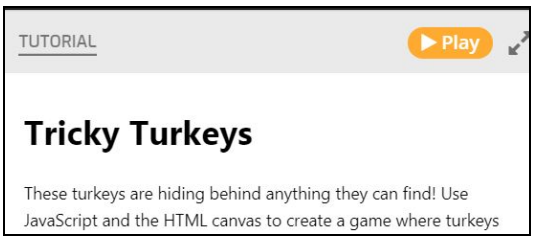
Students will practice reinforce JavaScript syntax as they create a fun Tricky Turkeys game! Students are also encouraged to expand on their project and modify their code. In this process, students will develop debugging and logical reasoning skills.

How do my students run their code?

Tell students to select the “play” button:

What does the tutorial include?

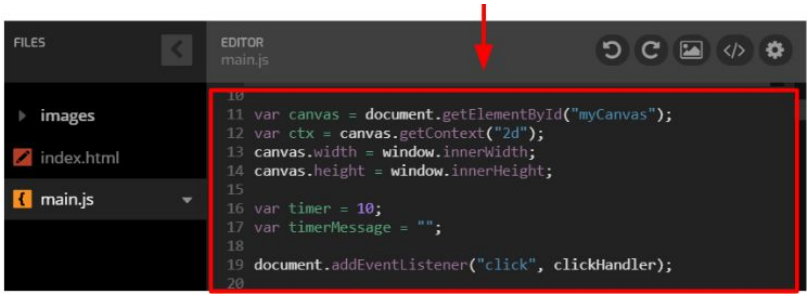
The tutorial includes several features and resources to help your students get started! Here’s what you will find:



- Step-by-step directions to help your students create a Tricky Turkeys project using JavaScript
- Bonus Challenges
- A “Helper Functions Reference Guide” that includes a list of functions with examples and descriptions.

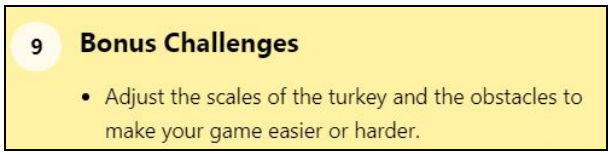
Where do my students edit their code?

This is the code editor section, where students can make changes to their code:



How do my students customize their project?

- Tell students to scroll down to the “Bonus Challenges” section of the tutorial, which includes directions to help your students make their project unique! Here’s what it looks like:



How can I help my students understand their code?

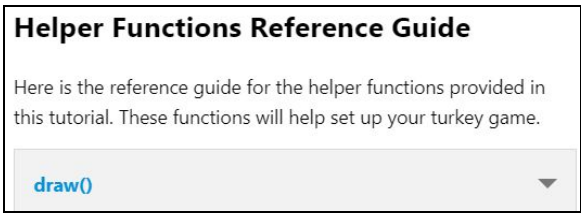
- Make sure students are reading the tutorial carefully. It describes each step of the Tricky Turkeys project.
- Tell students to read the comments (noted with the “//”) symbol. Here’s an example that tells students about the **draw()** function:

```
// draw()

////////////////////////////////////////////////////////////
//
// draw()
//
// This is the function used to update the canvas.
// First it fills in the background color and
// displays the timer text. Then, in order, it
// draws any hidden turkeys, draws all obstacles,
// and draws any found turkeys.
//
////////////////////////////////////////////////////////////
```

- If students need help with their syntax, ask them to...
  - Compare their code to the provided sample code. Here's an example:

  - Refer to the code in the “Helper Functions Reference Guide”:



- By the end of the project, their code for the Tricky Turkeys game could look similar to this (please see next page):

Tricky Turkeys: JavaScript

```
var canvas = document.getElementById("myCanvas");
var ctx = canvas.getContext("2d");
canvas.width = window.innerWidth;
canvas.height = window.innerHeight;

var timer = 10;
var timerMessage = "";

document.addEventListener("click", clickHandler);

var imgTurkey = new Image();
imgTurkey.src = "images/turkey.png";
imgTurkey.onload = draw;
var imgPumpkin = new Image();
imgPumpkin.src = "images/pumpkin.png";
imgPumpkin.onload = draw;

var turkeyFound = [0, 0, 0];
var turkeyGridX = [0, 0, 0];
var turkeyGridY = [0, 0, 0];
var turkeyScale = 0.1;

var obstacles = [imgPumpkin];
var obstacleScales = [0.1];

var gridScale = 100;
var topMargin = 50;

var turkeyGrid = [
  [0, 0, 0],
  [0, 0, 0],
  [0, 0, 0]
];

for (var i=0; i < turkeyFound.length; i++) {
  var tempX = 0;
  var tempY = 0;
  do {
    tempX = Math.floor(Math.random() * turkeyGrid.length);
    tempY = Math.floor(Math.random() * turkeyGrid[0].length);
  }
  while (turkeyGrid[tempX][tempY]);
  turkeyGrid[tempX][tempY] = 1;
  turkeyGridX[i] = tempX;
  turkeyGridY[i] = tempY;
}

var obstacleGrid = [
  [0, 0, 0],
  [0, 0, 0],
  [0, 0, 0]
];

for (var i=0; i < obstacleGrid.length; i++) {
  for (var j=0; j < obstacleGrid[0].length; j++) {
    obstacleGrid[i][j] = Math.floor(Math.random() * obstacles.length);
  }
}

timerMessage = "You have " + timer + " seconds to find " + turkeyFound.length + " turkeys!";
draw();
setInterval(updateTimer, 1000);

function clickHandler(event) {
  for(i = 0; i < turkeyFound.length; i++) {
   if(mouseIsOnTurkey(i)) {
      turkeyFound[i] = 1;
    }
  }

  draw();
}
```

Note: This is an open-ended project. Students are encouraged to experiment with their code, upload images, try the bonus challenges, explore the "Helper Functions Reference Guide", and expand on the provided code samples. For example, if students include extra uploaded obstacle images, part of their code might look like this:

```
var imgLeaf = new Image();
imgLeaf.src = "images/leaf.png";
imgLeaf.onload = draw;

var obstacles = [imgPumpkin, imgLeaf];
var obstacleScales = [0.1, 0.2];
```

How can I contact the Tynker support team?

If you have any issues or questions, send us an email at support@tynker.com.