Unit 4 - JavaScript Copied from: Introduction to Coding: Computer Programming, Copied on: 02/21/22

Content Area: **Sample Content Area**
Course(s): **Sample Course**
Time Period: **OctNov**
Length: **Sample Length & Grade Level**
Status: **Published**

INTRODUCTION TO JAVASCRIPT

Department of Curriculum and Instruction

Belleville Public Schools

Curriculum Guide

Introduction to Coding, Grades 9-12

INTRODUCTION TO JAVASCRIPT

Belleville Board of Education

102 Passaic Avenue

Belleville, NJ 07109

Prepared by: Teacher, Corey Woodring

Dr. Richard Tomko, Ph.D., M.J., Superintendent of Schools

Ms. LucyAnn Demikoff, Director of Curriculum and Instruction K-12

Ms. Nicole Shanklin, Director of Elementary Education

Mr. Joseph Lepo, Director of Secondary Education

Board Approved:

Unit Overview

What is JavaScript?

JavaScript is primarily known as the language of most modern web browsers, the Javascript language has continued to evolve and improve. JavaScript is a powerful, flexible, and fast programming language now being used for increasingly complex web development and beyond!

Since JavaScript remains at the core of web development, it's often the first language learned by coders eager to learn and build. Students will be able to create with the JavaScript foundation.

In this Unit, students will learn introductory coding concepts including data types and built-in objects—essential knowledge for all aspiring developers. This foundation for students will set you up for understanding the more complex concepts students will encounter later.

Enduring Understandings:

- Programming means giving instructions to a computer in a language it can understand.
- JavaScript is one programming language.
- The syntax of a programming language matters.
- The order of my code matters.
- Learning to code can influence my future.

Essential Questions

Essential Question: What does it mean to be a programmer?
How do computers keep track of information?

Exit Skills

Exit skills from learning /developing in Javascript

1. Identify the problem you need to solve
2. Break the problem down into smaller problems
3. Solve each small problem
4. Assemble your solutions into the final solution.

New Jersey Student Learning Standards (NJSLS-S)

CS.9-12.8.1.12.CS.2	Model interactions between application software, system software, and hardware.
CS.9-12.8.2.12.EC.3	Synthesize data, analyze trends, and draw conclusions regarding the effect of a technology on the individual, culture, society, and environment and share this information with the appropriate audience.
CS.9-12.CS	Computing Systems Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information. Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.

Interdisciplinary Connections

SCI.HS-ETS1-3	Evaluate a solution to a complex real-world problem based on prioritized criteria and trade-offs that account for a range of constraints, including cost, safety, reliability, and aesthetics, as well as possible social, cultural, and environmental impacts.
SCI.HS-ETS1-2	Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.
CAEP.9.2.12.C.1	Review career goals and determine steps necessary for attainment.
CAEP.9.2.12.C.2	Modify Personalized Student Learning Plans to support declared career goals.

Learning Objectives

Upon completion of this unit, students will be able to describe learning objectives which are programming concepts

here are some examples -

In this unit, students will be introduced to JavaScript as they complete engaging lessons, solve challenging puzzles, and build their own games in JavaScript. As they work through the unit, students gain a mastery of JavaScript syntax, as well as the ability to creatively program games and other projects and debug their own code.

Explaining and describing programming concepts in Javascript:

Variables

“Variables in Javascript are like boxes that hold things. You give them names, just like labeling a box of Pokémon cards, so you can find it quickly. Use good names, so you know exactly what’s inside—like ‘Ultimate Pokémon Deck’.”

Arrays

"A Javascript array is like your bookshelf (see header photo). It holds many different things in one place and makes them easy to access and find later. Your bookshelf has books, toys, drawings, photos, pencils, cards and more. If you ask me to find your Harry Potter book at the end of the bottom shelf, it would be easy."

Objects

"Most things around you are objects—people, cars, planes, trains. Objects have certain things that define who/what they are and things they can do. In Javascript, the things that give an object its identity are called 'properties' and the things they can do are called 'methods'. Think about your class made up of Student objects: some properties of a Student are: name, age, school, class. Some methods of a Student are: talk, run, and jump."

"Arrays can also hold objects. You can make an Array of your entire class that contains 30 Student objects then use a loop to print out all of the students in your class."

Functions

"Functions are like cooking recipes or the instructions that come with a LEGO set. Running a function goes through each of the steps until it reaches the end or is told to stop—like when we call you down for dinner!

A 'bake-a-chocolate-chip-cookie()' function includes all the steps required to bake a chocolate chip cookie—gathering and preparing ingredients, mixing, placing on baking tray, preheating oven and baking."

Parameters

"Parameters are like bits of information you share with your functions—giving them extra powers. Instead of having to write a separate 'bake' function for every type of cookie, you can write one 'bake()' function and tell it which type of cookie to bake as a parameter—bake('chocolate_chip_cookie')."

*Don't get too bogged down in the difference between parameters and arguments; it's not worth it at this point.

Closures

"A closure is like a dinosaur fossil—a snapshot from a moment in time preserved for millions of years. You can get still get information about the dinosaur from the fossil, even though the dinosaur itself has been gone for millions of years."

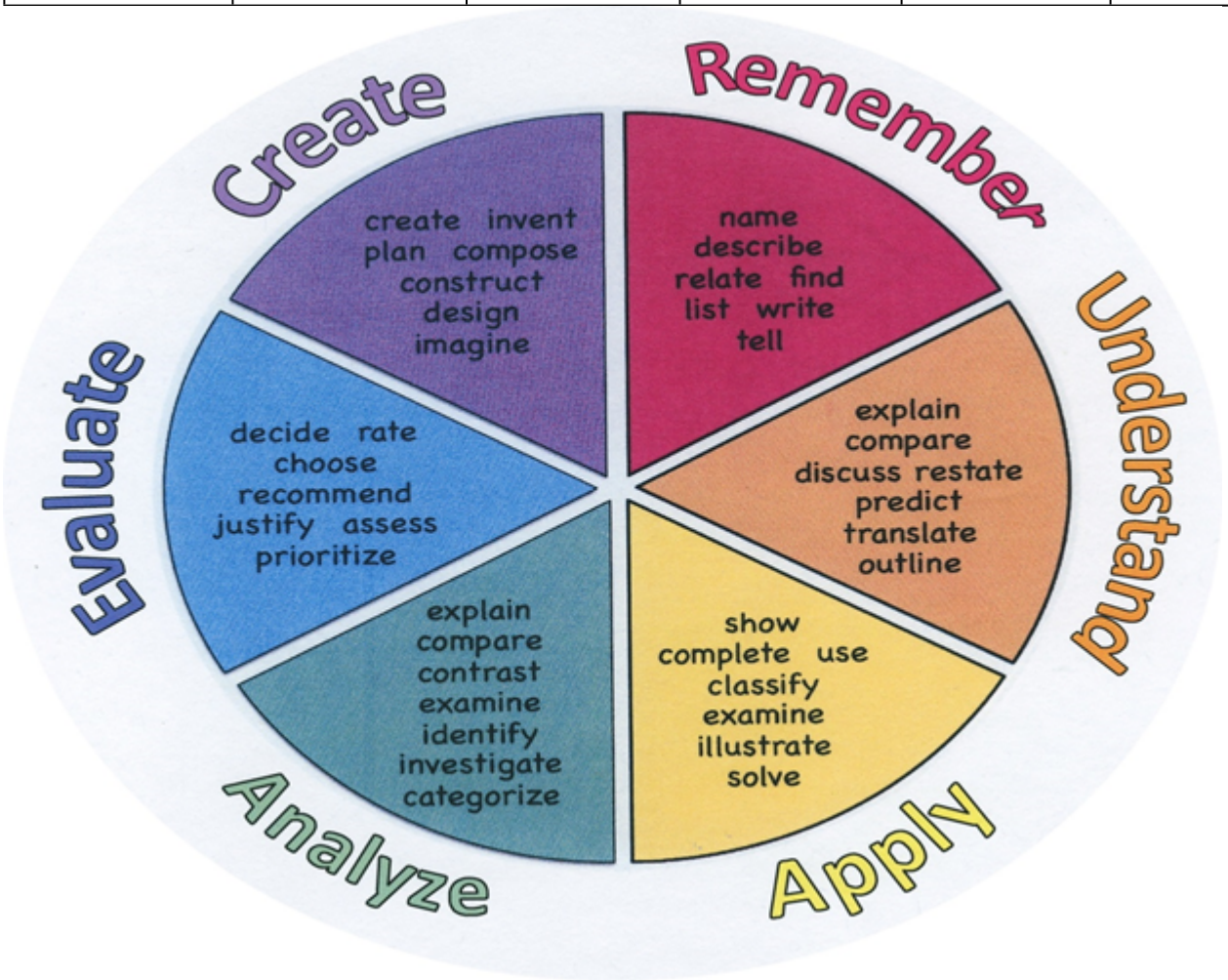
Closures were starting to make sense, so I tried using *cookies* again . . .

"Let's say I go to the same bakery every day and ask the baker for a cookie. The first thing he/she asks is, 'What kind of cookie do you want?' After a few days, the baker might already know what cookie I like and simply ask, 'The usual?'. By saving my cookie choice in memory, he/she is able to reuse the same function from the previous day to get me the proper cookie without having to ask again."

"So, it's like when we go to the barber and he just cuts our hair without asking what we want? He always asks new customers how they want him to cut, but never asks us anymore."

Remember	Understand	Apply	Analyze	Evaluate	Create
Choose	Classify	Choose	Categorize	Appraise	Combine
Describe	Defend	Dramatize	Classify	Judge	Compose
Define	Demonstrate	Explain	Compare	Criticize	Construct
Label	Distinguish	Generalize	Differentiate	Defend	Design
List	Explain	Judge	Distinguish	Compare	Develop
Locate	Express	Organize	Identify	Assess	Formulate
Match	Extend	Paint	Infer	Conclude	Hypothesize
Memorize	Give Examples	Prepare	Point out	Contrast	Invent
Name	Illustrate	Produce	Select	Critique	Make

Omit Recite Select State Count Draw Outline Point Quote Recall Recognize Repeat Reproduce	Indicate Interrelate Interpret Infer Match Paraphrase Represent Restate Rewrite Select Show Summarize Tell Translate Associate Compute Convert Discuss Estimate Extrapolate Generalize Predict	Select Show Sketch Solve Use Add Calculate Change Classify Complete Compute Discover Divide Examine Graph Interpolate Manipulate Modify Operate Subtract	Subdivide Survey Arrange Breakdown Combine Detect Diagram Discriminate Illustrate Outline Point out Separate	Determine Grade Justify Measure Rank Rate Support Test	Originate Organize Plan Produce Role Play Drive Devise Generate Integrate Prescribe Propose Reconstruct Revise Rewrite Transform



Suggested Activities & Best Practices

https://www.w3resource.com/javascript-exercises/
https://www.freecodecamp.org/learn/javascript-algorithms-and-data-structures/basic-javascript/

https://www.codecademy.com/learn/introduction-to-javascript

JavaScript is a cross-platform, object-oriented scripting language. It is a small and lightweight language. Inside a host environment (a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

JavaScript contains a standard library of objects, such as Array, Date, and Math, and a core set of language elements such as operators, control structures, and statements. Core JavaScript can be extended for a variety of purposes by supplementing it with additional objects.

The best way we learn anything is by practice and exercise questions.

Assessment Evidence - Checking for Understanding (CFU)

Formative Assessments

• Think, pair, share review questions from text.

• Practice mini-programs to strengthen concepts as taught.
• Teacher Observation

• Utilizing Gliffy.com to flowchart programs
Summative Assessments

• Chapter Test

• End of Chapter Projects from book.

example: https://quizlet.com/76598385/ap-computer-science-a-flash-cards/

Alternate Assessment

Web-based Assessment

Benchmark Assessment

- Admit Tickets
- Anticipation Guide
- Common Benchmarks
- Compare & Contrast
- Create a Multimedia Poster
- DBQ's
- Define
- Describe
- Evaluate
- Evaluation rubrics
- Exit Tickets
- Explaining
- Fist- to-Five or Thumb-Ometer
- Illustration
- Journals
- KWL Chart
- Learning Center Activities
- Multimedia Reports
- Newspaper Headline
- Outline
- Question Stems
- Quickwrite
- Quizzes
- Red Light, Green Light
- Self- assessments
- Socratic Seminar
- Study Guide
- Surveys
- Teacher Observation Checklist
- Think, Pair, Share
- Think, Write, Pair, Share

- Top 10 List
- Unit review/Test prep
- Unit tests
- Web-Based Assessments
- Written Reports

Primary Resources & Materials

Primary Resources:

Edhesive Online Training
Code.Org Online Training

Computers and Internet Access
AP Central at Collegeboard.org
Massive Open Online Course
Code.org
Multimedia Applications Tools
Abelson, H., Ledeen, K., and Lewis, H. R. Blown to Bits: your life, liberty, and happiness after the digital explosion. Upper Saddle River, N.J.: Addison-Wesley, 2008.

Ancillary Resources

Programming Resources:

Alice - This 3-D modeling environment allows students to create and animate 3-D worlds. This environment lends itself well to creating stories and games.

App Inventor - This open-source Web application allows students to create their own applications on mobile devices. App Lab - This is a programming environment for creating web applications with JavaScript. It allows students to develop programs and toggle back and forth between block-based and text-based programming modes.
EarSketch - This browser-based application allows students to create their own music using either JavaScript or Python. Greenfoot - This Java IDE is designed for use in education to create two-dimensional graphic applications, such as simulations and interactive games.
Java - There are several IDEs that can be used to write in Java. The Java language allows students to create and solve problems that vary widely in difficulty.
JavaScript - This language is commonly used to create interactive effects within Web browsers.
Lego Mindstorms NXT - This product integrates programming with Lego bricks and sensors to create and program robots. The instructions are assembled by linking together function blocks.
Processing - This programming language was initially created to serve as a software sketchbook, and it can be used to teach programming using a visual context.
Python - This language has the benefit of readability that might be helpful to new programmers.
Scratch - This blocks-based programming language allows students to build scripts to run animations. This product can be downloaded and installed on a computer or run in the browser.
Snap! - This Scratch-style programming language is block-based and allows users to define new primitives in JavaScript. Users can read and write information from the Internet using server-defined APIs and make mobile applications.
Swift - This programming language is designed for use with iOS, OS X, tvOS and watchOS. This environment allows students to

create their own Apple apps and includes interactive environments that allow students to see the effects of changes or additions to code as they type.

Design and Development Process:

What Is the Software Development Life Cycle?" Official Blog Airbrake Bug Tracker. https://airbrake.io/blog/insight/what-is-the-software-development-life-cycle

"Engineering Design Process." https://www.teachengineering.org/ engrdesignprocess.php
"The Engineering Design Process." http://www.eie.org/overview/engineeringdesign-process Mohammed, Nabil, Ali Munassar, and A. Govardhan.
"A Comparison Between Five Models of Software Engineering." IJCSI International Journal of Computer Science 7.5 (2010): 94-101.

Open Source:

"What Is Open Source?" Opensource.com. https://opensource.com/resources/whatopen-source

Open Source Initiative. http://opensource.org/

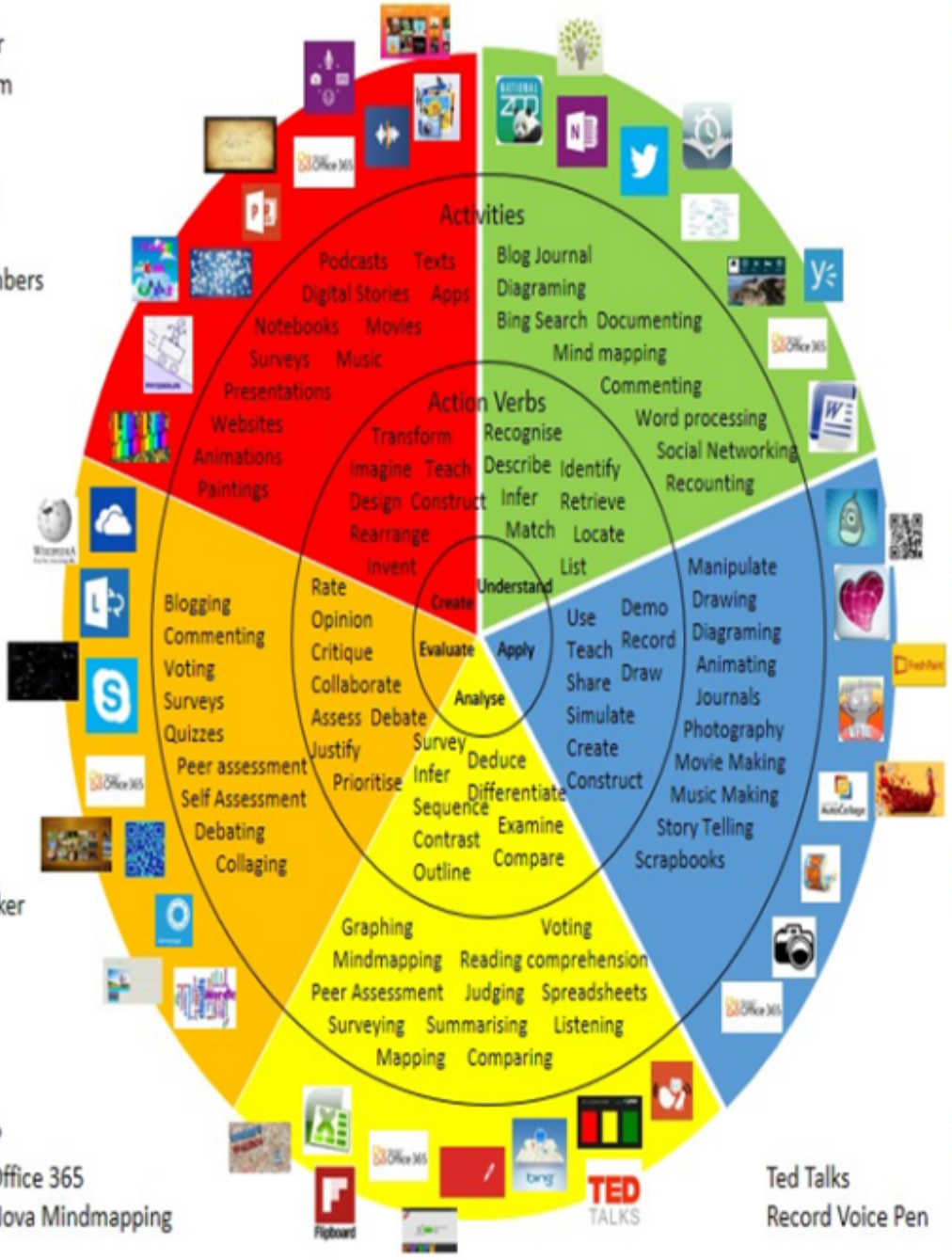
Technology Infusion

Technology Infusion and/or strategies include chromebooks online materials google/powerpoint slides

Win 8.1 Apps/Tools Pedagogy Wheel

Podcasts
Photostory 3
Kid Story Builder
Music Maker Jam
Paint A Story
Office 365
MS PowerPoint
Stack 'Em Up
NqSquared Numbers
Physamajig
Xylophone 8

Wikipedia
Skydrive
Lync
SkyMap
Skype
Office 365
Puzzle Touch
Easy QR
Memorylage
Life Moments
Word Cloud Maker



Where's Waldo?
MS Excel Office 365
Flipboard Nova Mindmapping

Ted Talks
Record Voice Pen

Originally taken from http://www.coetail.com/vzimmer/files/2013/02/iPadagogy-Wheel.001.jpg
And adapted for Windows 8.1 devices by Charlotte Beckhurst @CharBeckhurst

Alignment to 21st Century Skills & Technology

21st Century Life and Careers 9.2.12.C.1 Review career goals and determine steps necessary for attainment. 9.2.12.C.3 Identify transferable career skills and design alternate career plans. 9.2.12.C.5 Research career opportunities in the United States and abroad that require knowledge of world languages and diverse cultures. 9.2.12.C.6 Investigate entrepreneurship opportunities as options for career planning and identify the knowledge, skills, abilities, and resources required for owning and managing a business. 9.3.IT-PRG.1 Analyze customer software needs and requirements. 9.3.IT-PRG.2 Demonstrate the use of industry standard strategies and project planning to meet customer specifications. 9.3.IT-PRG.3 Analyze system and software requirements to ensure maximum operating efficiency.
9.3.IT-PRG.4 Demonstrate the effective use of software development tools to develop software applications.
9.3.IT-PRG.5 Apply an appropriate software development process to design a software application.
9.3.IT-PRG.6 Program a computer application using the appropriate programming language. 9.3.IT-PRG.7 Demonstrate software testing procedures to ensure quality products. 9.3.IT-PRG.8 Perform quality assurance tasks as part of the software development cycle.

Mastery and infusion of **21st Century Skills & Technology** and their Alignment to the core content areas is essential to student learning. The core content areas include:

- English Language Arts;
- Mathematics;
- Science and Scientific Inquiry (Next Generation);
- Social Studies, including American History, World History, Geography, Government and Civics, and Economics;
- World languages;
- Technology;
- Visual and Performing Arts.

21st Century Skills/Interdisciplinary Themes

Upon completion of this section, please remove all remaining descriptions, notes, outlines, examples and/or illustrations that are not needed or used.

Please list only the **21st Century/Interdisciplinary Themes** that will be incorporated into this unit.

- Communication and Collaboration
- Creativity and Innovation
- Critical thinking and Problem Solving

- ICT (Information, Communications and Technology) Literacy
- Information Literacy
- Life and Career Skills
- Media Literacy

CAEP.9.2.12.C	Career Preparation
CAEP.9.2.12.C.1	Review career goals and determine steps necessary for attainment.
CAEP.9.2.12.C.2	Modify Personalized Student Learning Plans to support declared career goals.

21st Century Skills

21st Century Skills that will be incorporated into this unit.

Critical Thinking & Problem Solving

Creativity and Innovation

Collaboration, Teamwork and Leadership

Cross-Cultural and Interpersonal Communication

Communication and Media Fluency

Accountability, Productivity and Ethics

- Civic Literacy
- Environmental Literacy
- Financial, Economic, Business and Entrepreneurial Literacy
- Global Awareness
- Health Literacy

Differentiation

• Technology Resources • Teacher Tutoring • Peer Tutoring • Cooperative Learning Groups • Differentiated Instruction • Follow all IEP Modifications/504 Plan

Exemplar: Teacher can use a pre-assessment to determine students' knowledge of standard being taught in lesson and then provide an extension activity for students

Differentiations:

- Small group instruction
- Small group assignments
- Extra time to complete assignments
- Pairing oral instruction with visuals
- Repeat directions
- Use manipulatives
- Center-based instruction
- Token economy
- Study guides
- Teacher reads assessments allowed
- Scheduled breaks
- Rephrase written directions
- Multisensory approaches
- Additional time
- Preview vocabulary
- Preview content & concepts
- Story guides
- Behavior management plan
- Highlight text
- Student(s) work with assigned partner
- Visual presentation
- Assistive technology
- Auditory presentations
- Large print edition
- Dictation to scribe
- Small group setting

Hi-Prep Differentiations:

- Alternative formative and summative assessments
- Choice boards
- Games and tournaments
- Group investigations
- Guided Reading
- Independent research and projects
- Interest groups
- Learning contracts
- Leveled rubrics
- Literature circles
- Multiple intelligence options
- Multiple texts
- Personal agendas
- Project-based learning
- Problem-based learning
- Stations/centers
- Think-Tac-Toes
- Tiered activities/assignments
- Tiered products

- Varying organizers for instructions

Lo-Prep Differentiations

- Choice of books or activities
- Cubing activities
- Exploration by interest
- Flexible grouping
- Goal setting with students
- Jigsaw
- Mini workshops to re-teach or extend skills
- Open-ended activities
- Think-Pair-Share
- Reading buddies
- Varied journal prompts
- Varied supplemental materials

Special Education Learning (IEP's & 504's)

Special Education Learning adaptations that could possibly be employed in the unit, using the ones identified below.

Exemplar -Adapting existing materials, simplifying or supplementing materials

- printed copy of board work/notes provided
- additional time for skill mastery
- assistive technology
- behavior management plan
- Center-Based Instruction
- check work frequently for understanding
- computer or electronic device utilizes
- extended time on tests/ quizzes
- have student repeat directions to check for understanding
- highlighted text visual presentation
- modified assignment format
- modified test content

- modified test format
- modified test length
- multi-sensory presentation
- multiple test sessions
- preferential seating
- preview of content, concepts, and vocabulary
- Provide modifications as dictated in the student's IEP/504 plan
- reduced/shortened reading assignments
- Reduced/shortened written assignments
- secure attention before giving instruction/directions
- shortened assignments
- student working with an assigned partner
- teacher initiated weekly assignment sheet
- Use open book, study guides, test prototypes

English Language Learning (ELL)

Exemplar:

*provide additional wait time for student responses to questions

When asked a question, ELL students typically translate it into their first language, formulate an answer in their first language, and translate an approximation of the answer into English, before giving their response.

- teaching key aspects of a topic. Eliminate nonessential information
- using videos, illustrations, pictures, and drawings to explain or clarif
- allowing products (projects, timelines, demonstrations, models, drawings, dioramas, poster boards, charts, graphs, slide shows, videos, etc.) to demonstrate student's learning;
- allowing students to correct errors (looking for understanding)
- allowing the use of note cards or open-book during testing
- decreasing the amount of workpresented or required
- having peers take notes or providing a copy of the teacher's notes
- modifying tests to reflect selected objectives
- providing study guides
- reducing or omitting lengthy outside reading assignments
- reducing the number of answer choices on a multiple choice test
- tutoring by peers
- using computer word processing spell check and grammar check features
- using true/false, matching, or fill in the blank tests in lieu of essay tests

At Risk

Exemplar:

Caring, Sustained Relationships

One of the shortcomings of our educational structure is that relationships with teachers, especially in secondary school, may be caring, but they are not easy to sustain. Yet at-risk youth need relationships that are both caring and stable. They need to build a sense of trust and have the time to communicate the complexity, frustrations, and positive aspects of their lives in and out of school. Only after creating a strong relational base will an adult have the platform to be a source of enduring and cherished advice to a student. Students won't confer trust to an adult based on his or her role as a counselor, psychologist, or social worker. We have to earn it by building a relationship.

Possible Intervention Strategies that will be employed in the unit, using the ones identified below.

- allowing students to correct errors (looking for understanding)
- teaching key aspects of a topic. Eliminate nonessential information
- allowing products (projects, timelines, demonstrations, models, drawings, dioramas, poster boards, charts, graphs, slide shows, videos, etc.) to demonstrate student's learning
- allowing students to select from given choices
- allowing the use of note cards or open-book during testing
- collaborating (general education teacher and specialist) to modify vocabulary, omit or modify items to reflect objectives for the student, eliminate sections of the test, and determine how the grade will be determined prior to giving the test.
- decreasing the amount of workpresented or required
- having peers take notes or providing a copy of the teacher's notes
- marking students' correct and acceptable work, not the mistakes
- modifying tests to reflect selected objectives
- providing study guides
- reducing or omitting lengthy outside reading assignments
- reducing the number of answer choices on a multiple choice test
- tutoring by peers
- using authentic assessments with real-life problem-solving
- using true/false, matching, or fill in the blank tests in lieu of essay tests
- using videos, illustrations, pictures, and drawings to explain or clarify

Talented and Gifted Learning (T&G)

Exemplar: Provide students with problem-based learning activity using multiple standards from the unit.

http://www.grandviewlibrary.org/CurriculumAdaptations/General_Gifted.pdf

Grouping • Group gifted students with other gifted students or higher-level learners. • Refrain from grouping gifted students with lower-level students for remediation.

Talented and Gifted adaptations that will be employed in the unit, using the ones identified below.

- Above grade level placement option for qualified students
- Advanced problem-solving
- Allow students to work at a faster pace
- Cluster grouping
- Complete activities aligned with above grade level text using Benchmark results
- Create a blog or social media page about their unit
- Create a plan to solve an issue presented in the class or in a text
- Debate issues with research to support arguments
- Flexible skill grouping within a class or across grade level for rigor
- Higher order, critical & creative thinking skills, and discovery
- Multi-disciplinary unit and/or project
- Teacher-selected instructional strategies that are focused to provide challenge, engagement, and growth opportunities
- Utilize exploratory connections to higher-grade concepts
- Utilize project-based learning for greater depth of knowledge

Sample Lesson

Javascript- Computer Science Course

Introduction to JavaScript

10.1: Intro to JS

Materials: Presentation (Included), Syntax Scavenger Hunt (Included), Syntax Scavenger Hunt Key (Included), Simple Bunny Game Code Handout (Included), Simple Bunny Game (Click "View Code" on the top right to remix)

1. Walk In: Get your colored pencils and fill out the top of your Syntax Scavenger Hunt handout.
2. Go over vocab from top of Syntax Scavenger Hunt
3. JavaScript Presentation

4. Start Scavenger Hunt using code handout
   **Note: I tell the students not to worry about getting everything right, but to see how much they can figure out. Some classes will need some of the next class to finish or to go over it

5. Show them the game

10.2: Explaining JavaScript

Materials: Explaining Bunny Code (Included), Explaining Bunny Code Key (Included), Simple Bunny Game (Click "View Code" on the top right to remix)

1. Walk In: Finish Scavenger Hunt

2. Go over scavenger hunt
   Note: I don't re-annotate the code as a class. For each section, I go over the answers to the questions and explain one or two examples in the code.

3. Share the Bunny Game code with the students, show them how to remix it (Click "View Code" on the top right) and how to comment out lines of code in case it helps them figure out what certain sections of code do.
   (**Note: only a few students actually use this technique, so if you have a distractible group, you might want to have them explain the code before they access it. Otherwise some of them get tempted to start modding the code before they have explained how it works)

4. Students work on explaining code and get it checked before they can move on

10.4: Modding

Materials: Bunny Mod Challenges (Included), Bunny Mod Challenges Key (Included)

1. Walk In: Pair up and finish explaining your code

2. What does modding mean? Modding is short for modifying code. It is popular in the video gaming world.

3. Students get their code explanations checked, then start working on the Bunny Mod Challenges. They should explain how they achieved each mod. When they finish, they can come up with their own extensions.

10.5: Continue Codding

10.6: Finish Modding

1. Walk In: What can you do in JavaScript that you couldn't in Scratch?
2. Share out
3. Finish modding
4. Gallery Walk

Note: At this point, if I have time, I often give students a week or so to work through some of Code.org's CS Discoveries curriculum. Since they have already learned the computational concepts, I make a packet of the concept overview pages from the CS Discoveries curriculum so they can read through and record how to do some of the things they learned in Scratch in JavaScript. Then I give them certain lessons to complete on Code.org to get them ready for the video game project. I have included my handout that guides them through sections of the CS Discoveries curriculum, but I have not included the concept overview pages as they are not mine to share.