

# DWE Approved Introduction to Object Oriented Programming Curriculum Content Frameworks

**Please note: All assessment questions will be  
taken from the knowledge portion of these  
frameworks.**

*Prepared by*

Marilyn Carrell, Springdale High School

*Facilitated by*

Karen Chisholm, Education and Instruction Manager  
Lesia Edwards, Education and Instruction Coordinator  
Office of Assessment and Curriculum  
Arkansas Department of Career Education

*Edited by*

James Brock, Education and Instruction Manager  
Tim Johnston, Education and Instruction Coordinator  
Ginger Fisher, Education and Instruction Coordinator  
LaTrenda Jackson, Education and Instruction Coordinator  
Peggy Wakefield, Education and Instruction Coordinator  
Office of Business and Marketing Technology  
Arkansas Department of Career Education

*Disseminated by*

Career and Technical Education  
Office of Assessment and Curriculum  
Arkansas Department of Career Education

# Curriculum Content Frameworks

## DWE Approved Introduction to Object Oriented Programming

Grade Levels: 9, 10, 11, 12  
Course Code: 492680

Prerequisite: Keyboarding and Algebra I

Course Description: This is a one-semester course designed as an introduction to the concepts of object oriented programming. It is taught in the language *Alice* and allows students to write on-screen "movies" and "games" in a 3D world. Students will learn programming concepts, such as if-then, loops, variable, methods, functions, parameters, and events in a fun and motivating environment.

### Table of Contents

	Page
Unit 1: Introduction to Alice Environment and Objects	1
Unit 2: Program Design and Programming in Alice	2
Unit 3: Writing Methods and Creating New Classes	3
Unit 4: Events and Event Handling	4
Unit 5: Functions and If/Else	5
Unit 6: Repetition with Loops	6
Suggested Supplemental Materials/Projects	7
Glossary	8

# Unit 1: Introduction to Alice Environment and Objects

## Hours: 10

Terminology: 2D, 3D, 3D Text, Animation, Billboard, Bounding box, Center point, Class, Computer program, Distance, Flowchart, Object, Object Oriented Programming (OOP), Orientation, Position, Six degrees of freedom, Viewpoint, Virtual world

<b>CAREER and TECHNICAL SKILLS</b>			
What the Student Should Know		What the Student Should be Able to Demonstrate	
<b>Knowledge</b>		<b>Application</b>	
1.1	Define terminology	1.1.1	Prepare a list of terms with definitions
1.2	Learn to open and play worlds	1.2.1	Open sample worlds and tutorials, using the play and restart buttons
1.3	Identify the parts of the Alice environment	1.3.1	Using the examples, find the toolbar, World View window, object tree, details panel, method editor, and events editor
1.4	Identify the parts of the world	1.4.1	Create a world. Place objects in the world and manipulate the properties of an object (such as color or opacity)
1.5	Explain 2D and 3D, center points, bounding boxes, and 6 degrees of freedom, orientation, position, and distance between objects	1.5.1	Access the bounding box, finding the center of an object, the x, y and z axes and identifying the orientation and position of the objects
1.6	Explain how the camera can be moved in the world and its viewpoint	1.6.1	Move the camera through the world so that it views the object from the front, back, and other views
1.7	Explain how the mouse mode buttons (move freely, move up and down, turn left and right, turn forward and backward, tumble, resize, and copy) can be used to modify objects	1.7.1	Use the mouse mode buttons to manipulate objects
1.8	Explain how the object tree can be used to identify and manipulate the subparts of an object	1.8.1	Use the mouse mode buttons to manipulate the subparts of an object
1.9	Explain the uses of the single view and quad view modes in the screen editor	1.9.1	Use single view to place an object and then quad view fine tune to the location of objects
1.10	Explain 3D-Text, how it is created, and how it is used	1.10.1	Write an Alice world that uses 3D-Text
1.11	Explain how billboards are different from other objects and how they are used	1.11.1	Write an Alice world that uses a billboard

## Unit 2: Program Design and Programming in Alice

### Hours: 10

**Terminology:** Algorithm, Argument (duration, style, asSeenBy), Bug, Comment, Do in order, Do together, Debug, Documentation, Instruction, Methods, Move, Turn, Roll, Resize, Say, Think, Play sound, Move to, Move toward, Move away from, Orient to, Point at, Turn to face, Set point of view to, Nesting, Program, Pseudocode, Property (vehicle), Runtime, Scenario, Sequential, State, Storyboard, Simultaneous, Syntax, Trial-and-error

<b>CAREER and TECHNICAL SKILLS</b>			
What the Student Should Know		What the Student Should be Able to Demonstrate	
<b>Knowledge</b>		<b>Application</b>	
2.1	Define terminology	2.1.1	Prepare a list of terms with definitions
2.2	List the steps in creating a 3-D animation program: read the scenario, design, implement, test	2.2.1	Read a scenario and accurately determine the requirements of the problem.
		2.2.2	Create a storyboard of the problem
		2.2.3	Implement the problem
		2.2.4	Test the problem
2.3	Explain the process of following a storyboard to implement sequential instructions	2.3.1	Implement a series of sequential instructions
2.4	Explain the difference between and usage of the control structures: Do in order and Do Together	2.4.1	Implement instructions that use Do together and Do in order
2.5	Explain how the term "nested" applies to Do in order and Do Together	2.5.1	Write a program that uses these instructions and then highlight the nesting on a printout
2.6	Explain the action performed by and how to use the primitive, built-in methods-particularly move, turn, roll, say, think, orient to, turn to face, point at, move to	2.6.1	Write programs that use these methods
2.7	Explain how arguments are used with methods and what common arguments specifies (duration, style--gently, abruptly, begin gently, end gently, as seen by) and how to use trial and error to tweak the arguments	2.7.1	Write methods using at least the listed arguments to control animation
		2.7.2	Use trial-and-error strategy to "tweak" the arguments to attain appropriate motion for an object
2.8	Explain the purpose for comments	2.8.1	Write programs that use comments appropriately
2.9	Explain purpose and uses of of vehicle, color, opacity, and isShowing properties	2.9.1	Create worlds in which these properties are changed at design time
2.10	Explain the difference in setting a property at design time and changing it in	2.10.1	Change a property of an object at runtime

## Unit 3: Writing Methods and Creating New Classes

### Hours: 10

**Terminology:** Abstraction, Boolean, By default, Calling a method, Class, Class file, Class-level method, Inheritance, isShowing property instance, Instantiate, Method, Naming conventions (TitleCase, camelCase, PascalCase), Object, Opacity property, Parameter, Primitive method, Stepwise refinement, String, World file, World-level method

<b>CAREER and TECHNICAL SKILLS</b>			
What the Student Should Know		What the Student Should be Able to Demonstrate	
<b>Knowledge</b>		<b>Application</b>	
3.1	Define terminology	3.1.1	Prepare a list of terms with definitions
3.2	Explain the naming conventions for classes, objects, methods, functions, and variables	3.2.1	Write programs where classes, objects, methods, functions, and variables are named following the conventions
3.3	Explain the difference in a class and an object and how a class is used to instantiate an object	3.3.1	Create a multiple instances of a class and change a property (such as color) of the class so that each object differs from the others
3.4	Explain how the use of world-level methods implements abstraction and how to create and call a method	3.4.1	Write world methods that allow each step on the storyboard to be performed
		3.4.2	Write the World.my first method to call these methods
3.5	Explain how parameters are used to communicate with methods and use of arguments in calling methods with parameters	3.5.1	Write methods that use parameters
3.6	Explain the different types of parameters: number, boolean, object, other (string)	3.6.1	Identify and use the correct data type for parameters
		3.6.2	Write methods and/or functions that use different types of parameters and methods with multiple parameters
3.7	Explain the difference between a world-level method and a class-level method	3.7.1	Write class-level methods, some of which use parameters
3.8	Explain the importance of object parameters in class-level methods	3.8.1	Write class-level methods that use object parameters
3.9	Explain the difference in saving a world and saving an object as a class	3.9.1	Identify world files and class files by their extensions, open and use each appropriately
3.10	Explain the reason for saving an object with custom methods as a class and how the new class inherits the properties and methods of the original class	3.10.1	Write a custom method for an object, save it as a class, and import the new class in another world
3.11	Explain the relationship between opacity and isShowing properties	3.11.1	Write programs that use the opacity and isShowing properties
		3.11.2	Write programs where one object "flies" or rotates around an invisible object

## Unit 4: Events and Event Handling

### Hours: 10

Terminology: Control of flow, Event, Event handling method, Event driven programming, Hebuilder/shebuilder, Incremental development, Interactive, User input

CAREER and TECHNICAL SKILLS			
What the Student Should Know		What the Student Should be Able to Demonstrate	
Knowledge		Application	
4.1	Define terminology	4.1.1	Prepare a list of terms with definitions
4.2	Explain the difference in the control of flow in an interactive program and a non-interactive program	4.2.1	Using example worlds, determine which are interactive worlds and which are non-interactive world
		4.2.2	Using a sample interactive world, identify the interactive events and event-handling methods
4.3	Explain the process of writing methods to respond to events and linking the events to the methods	4.3.1	Write worlds with method(s) to respond to events and link the event(s) to the method(s)
4.4	Explain the process of testing and incremental development and why events are world level	4.4.1	Write a world with event handling methods using the incremental development method
4.5	Give examples of how using parameters with event handling methods allows for re-use of the event	4.5.1	Write a world which uses one event handling method to respond to multiple events.
4.6	Explain why it is important to test with different situation and give examples of how to test a method with a numeric parameter	4.6.1	Test a world
4.7	Explain the features of and process of using hebuilder/shebuilder	4.7.1	Use hebuilder or shebuilder to create people and use them in a world

## Unit 5: Functions and If/Else

### Hours: 10

**Terminology:** Boolean expression, Conditional execution, Conditional expression, Expression, Function, If/Else statement, Integer, Logical operator, Random numbers, Range, Relational operator, Return statement

<b>CAREER and TECHNICAL SKILLS</b>	
What the Student Should Know	What the Student Should be Able to Demonstrate
<b>Knowledge</b>	<b>Application</b>
5.1 Define terminology	5.1.1 Prepare a list of terms with definitions
5.2 Explain the difference in a function and a method	5.2.1 Write programs that use functions
5.3 Explain what an identifier is, the data types, and naming conventions	5.3.1 Select data types for sample data items
	5.3.2 Select appropriate identifiers following naming conventions
5.4 Explain the process of writing a custom function	5.4.1 Write custom functions
	5.4.2 Write programs that use the custom functions
5.5 Explain the return statement	5.5.1 Write functions that return a number
	5.5.2 Write functions that return a Boolean
5.6 List the relational operators and logical operators and evaluate Boolean expressions	5.6.1 Write Boolean expressions using the relational operators
	5.6.2 Write Boolean expressions using both relational operators and logical operators
5.7 Explain how If statements use boolean expressions and when the Else should be used and evaluate the flow of control in sample If/Else statements	5.7.1 Write programs that use If/Else statements
5.8 Explain how nested if can be used to handle situations where there are three or more alternatives and evaluate the flow of control in examples	5.8.1 Write programs that use nested If/Else statements
5.9 List several uses for random numbers, features of the random number function, and the need for integer values in some cases	5.9.1 Write random statements that will return a number within a particular range
	5.9.2 Write programs that use random numbers to control the action

## Unit 6: Repetition with Loops

### Hours: 10

Terminology: BDE (Begin-During-End) event, Count, Definite loop, Indefinite loop, Infinite loop, Nested loops, Loop, While

<b>CAREER and TECHNICAL SKILLS</b>			
What the Student Should Know		What the Student Should be Able to Demonstrate	
<b>Knowledge</b>		<b>Application</b>	
6.1	Define terminology	6.1.1	Prepare a list of terms with definitions
6.2	Explain why Loop statement is called a counted loop and a definite loop and how to use the Loop structure	6.2.1	Write programs that use Loop statements
6.3	Explain how Loop statements can be used to create nested loops	6.3.1	Write programs that use nested Loop statements
6.4	Describe a situation in which an infinite loop should be used	6.4.1	Write a program that uses an infinite loop
6.5	Explain why a While is described as a conditional loop or indefinite loop and evaluate examples	6.5.1	Write programs that use While loops
6.6	Explain how BDE is used with a While something is true event	6.6.1	Write programs that use a While something is true and use BDE

## Suggested Supplemental Materials/Projects

A final project is strongly recommended. The student should design and implement his or her own program, which he or she has designed. This allows for creativity, pride in work, and an opportunity to put all skills together in a meaningful way.

Do some simple recursive problems from your textbook. Having been introduced to recursion in a visible manner will be hugely beneficial to your student in higher level programming courses.

Introduce your students to Lists (or Lists and Arrays). They will like Lists: they are easy to use and give a lot of power with simple commands. The real benefit will be that your students will be much better prepared for these topics in higher level programming courses.

Alice is a free programming language developed at Carnegie Mellon University and is available for download at [Alice.org](http://Alice.org). These frameworks were developed using Alice version 2.2.

# Glossary

## Unit 1: Introduction to Alice Environment and Objects

1. 2D (Two dimensional) – objects that have only two dimensions, height and width. They are flat, can only be moved side to side or up and down, and can be viewed from the front
2. 3D (Three dimensional) – objects that have three dimensions: height, width, and depth. They can be moved left or right, up or down, forward or backward and can be seen from any side
3. 3D Text – a 3D object where the user supplies the text
4. Animation – an illusion of movement created when the scene is drawn with objects, then redrawn with the objects in a slightly different place, over and over again
5. Billboard – a flat, 2D image in a world. It can be any gif, jpg, or tif. It is often used for instructions or credits
6. Bounding box – a yellow box that highlights the selected object.
7. Center point – the point in an object where the x (left, right), y (up, down) and z (forward, backward) axes meet. It is not necessarily at the middle of the object. It is the rotation point selected by the designer of the object.
8. Class – a set of specifications that describes a particular type of object. The class can be compared to a blueprint, which are the plans for the house--not the house itself. Classes are located in the Alice Gallery
9. Computer program – a set of instructions that tells a computer what to do
10. Distance – measured from the center of one object to the center of another object
11. Flowchart – a diagram of the logic of a program constructed using a basic collection of symbols
12. Object – an instance of a class created in the world. For example, Alice creates an object (a rabbit) in the world using the instructions found in the class-Rabbit located in the Gallery. Each object has properties, methods, and functions
13. Object Oriented Programming (OOP) – a type of computer programming in which programming objects are used to form additional objects
14. Orientation – direction an object is facing
15. Position – the center of the object is its position in the world. (This along with its orientation can be viewed in the Point Of View Property)
16. Six degrees of freedom- the six directions an object can move in a 3D world: up, down, left, right, forward, backward
17. Viewpoint – the camera's position, including the direction in which it is pointed
18. Virtual world – a program, simulation, or video game implemented in a 3D

## Unit 2: Program Design and Programming in Alice

1. Algorithm – a list of steps needed to solve a problem
2. Argument – an item of information that must be supplied so that Alice can execute a method. The values that are sent in to match the parameters are arguments (See Unit 3)
3. asSeenBy – tells Alice to use the orientation of one object to guide the movement of another object
4. Bug – an error in a computer program
5. Comment – remarks are explanation lines within a program. In Alice and Java, comments begin with //. Comments are one form of documentation
6. Debug – find and correct errors in a computer program
7. Do in order – control statement that tells Alice to do a block of statements sequentially
8. Do together – control statement that tells Alice to do a block of statements simultaneously
9. Documentation – explanatory material about the program coding and how to use the program. Manuals, readMe files, help screens, and comments are all forms of documentation
10. Instruction – a single "command" in the program code
11. Method – a named program segment (a small set of instructions) that defines how to perform a task
12. Move – generic method where the programmer specifies the distance and the direction
13. Move away from – method that causes the object to move the specified distance away from another object
14. Move to – method that causes the object to move to another object's center point. At the conclusion of the move, the center points of both objects will be in the same location
15. Move toward – method that causes the object to move the specified distance toward another object
16. Nesting – a programming statement written within another
17. Orient to – method that causes the object to point in the same direction as another object. The Up, Right, and Forward axes will be aligned with the specified object
18. Play sound – method that is used to play any MP3 or WAV file or one of the sounds Alice provides
19. Point at – method similar to turn to face, except the object will be tilted so its forward axis is point at the other object's center

20. Program – set of instructions that tells the computer what to do
21. Property – values that specify the object's characteristics
22. Pseudocode – set of instructions in a combination of English and programming language that will be eventually be converted into the programming language
23. Resize – method that changes the size of the object by a specified amount
24. Roll – method that causes the object to (rotate on the y axis left or right) from its center point
25. Runtime – the time while the animation is running
26. Say – method that displays a cartoon-like speech bubble that contains the text the programmer specifies for the message
27. Scenario – a problem statement that describes the overall animation. It tells what problem is to be solved or lesson is to be learned
28. Sequential – to move in order from one statement to another. Do in order is sequential. Unless instructed otherwise, computer execute the instruction sequentially
29. Set point of view to – method that sets the point of view to the point of view of another object. It is frequently used with the camera (along with move to) to let the camera film what the object is "seeing"
30. Simultaneous – to perform several statements at the same time. Do together is an example of simultaneous execution
31. State – snapshot of a scene in the animation
32. Storyboard – a visual design method that breaks the program down into sequence of major scenes with transitions between scenes. It includes a sketch, description, list of files needed, and text
33. Style argument – specifies the way in which one movement instruction blends into the next. The options are gently (begins and ends gently), abruptly (begins and ends abruptly), begin gently (begins gently, ends abruptly), end gently (begins abruptly and ends gently)
34. Syntax – the grammar of a language; statement structure and punctuation
35. Trial-and-error – a strategy where the programmer tries different amounts until he/she finds one that works best
36. Turn to face – method that causes an object to turn to face another object
37. Vehicle – a property that allows the object A to be "coupled" with Object B. That way when Object B moves, Object A moves along with it

## Unit 3: Writing Methods and Creating New Classes

1. Abstraction – a process of knowing the general concept, or what an item does, without knowing all the details of how it was implemented. Using a method is an example of abstraction. This allows the programmer to think about the general task without having to worry about all the small steps that were needed to complete the task
2. Boolean – data type that has one of two values, true or false
3. By default – the preset value that will be used if another value is not supplied
4. Calling a method – causes Alice to animate the instructions within the method
5. Class-level method – methods that define behaviors for a single object
6. Inheritance – creating a new class that has additional features in addition to all the old methods, functions, and properties of the original class
7. isShowing property – Boolean property that makes an object visible (true) or invisible (false)
8. Instance – an object. Every object is an instance of a class; it was created from the class instructions or "blueprint"
9. Instantiate – the creation of an object. When an object is created and displayed, this is instantiating the class
10. Method – named set of instructions that when carried out will perform a task
11. Naming conventions – a standard way of naming identifiers used by programmers to help designate the exact function of the identifier
12. PascalCase – a naming convention used for class names where the first letter of each word is capitalized--with no spaces. For example, DancingGirl
13. camelCase – a naming convention used for variables, methods, and functions where the first letter is lowercase, the first letter of following words is capitalized. For example, slideLeft
14. Object – an instance of a class. When the programmer double clicks on a class, he/she adds an object--an instance of the class--to the world.
15. Opacity property – a property that determines how opaque (how hard to see through) with 0% being transparent and 100% being solid
16. Parameter – a method or function variable that acts as a basket to receive information that is sent to the method or function. Parameters can be Number, Boolean, Object, or Other types
17. Primitive method – the built-in methods that come with all Alice objects. Some examples are move, turn, and roll
18. Stepwise refinement – a design technique that breaks the overall task down into abstract tasks and then breaks each task down into smaller pieces and then define the steps in each piece
19. String – a data type that is text--a string of characters
20. World-level method – methods that specifically reference more than one object

## Unit 4: Events and Event Handling

1. Control of flow – the sequence of actions of a program. In an interactive program, sequence of actions is dependent on what the user does. In a noninteractive program, the sequence of actions is determined by the programmer
2. Event – something that happens, like a mouse click or key press
3. Event handling method – a method, linked to an event, that has instructions that are the response to an event
4. Event driven programming – a programming technique that responds to user events; examples include Alice interactive programs and Visual Basic
5. Hebuilder/shebuilder – two special Alice people-building options, where the programmer can select the desired body type, hair, skin color, eyes and clothing
6. Incremental development – a recommended programming development strategy where each method is written and tested and corrected, then the next method is done, until the program is finished. This approach allows the programmer to fix a problem before it causes issues elsewhere
7. Interactive – a program that requires user input--mouse click, pressing a key, etc.
8. User input – keying, pressing arrow keys, moving the mouse, clicking the mouse, and other actions that the user does to indicate his responses

## Unit 5: Functions and If/Else

1. Boolean expression – an expression that uses relational operators (and possibly logical operators) to determine a true or false value. Example, `age >= 16`
2. Conditional execution – the use of If/Else to branch to different segments of code
3. Expression – a phrase that uses operators and evaluates to a result, often numeric or Boolean
4. Function – named set of instructions that returns a value
5. If/Else statement – a control structure that makes decision based on a Boolean expression and transfers control to that section. If the value is true, then the action is transferred to the If part; if it is false, it is transferred to the Else part
6. Integer – positive or negative whole number. If `integerOnly` argument is chosen the random number function will return whole numbers
7. Logical operator – operators that are used to connect or modify Boolean expressions. In Alice, the logical operators are `not`, `and`, `or`, and `both`
8. Random numbers – a number from a specified range chosen without a pattern. If no range is specified, the random number will be a fractional value between 0 and 1
9. Range – a set of values, often defined by its minimum and maximum
10. Relational operator – used to compare values in Boolean expressions: `==`, `!=`, `>`, `>=`, `<`, and `<=`
11. Return statement – a required statement in a function which is used to send the result back to the calling instruction

## Unit 6: Repetition with Loops

1. BDE (Begin-During-End) event – While events; the programmer may specify what is to happen as the event begins, during, and ends
2. Count – describes the number of times a loop repeats; a Loop statement is a counted loop
3. Definite loop – a control statement that can be used when we know how many to repeat a block of instructions; Loop is a definite loop
4. Indefinite loop – a control statement that can be used when we do not know exactly how many times to repeat a block of instructions. A While statement is an indefinite loop
5. Infinite loop – a loop that continues until the program stops
6. Nested loops – a Loop statement within a Loop statement
7. Loop – a control statement that can be used when we know how many times to repeat a block of instructions; a definite loop
8. While – a control statement that can be used when we do not know exactly how many times to repeat a block of instructions; an indefinite loop