

Introduction to **GAME SYSTEMS DESIGN**



Dax **GAZAWAY**

Introduction to Game Systems Design

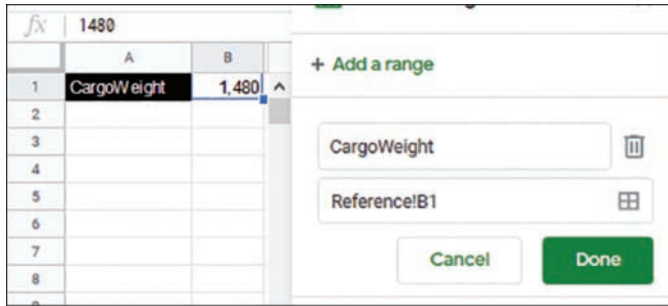


Figure 5.54 Named range cell

fx	=CargoWeight*3	
	A	B
1	CargoWeight	1,480
2		4440

Figure 5.55 A named range in a formula

Keep in mind the following important notes about named ranges:

- Named ranges can be used from anywhere in a workbook (and so they are *global*), and they can be accessed from any sheet without requiring a reference to the sheet itself. So, for example, you can access `MonsterNames` and `CargoWeight` anywhere in the workbook by simply using those names.
- Named ranges are dynamic. If the contents of the cells they contain change, then anything referencing the range will also change. The same exception for drop-down validation also applies to named ranges: The list changes in the drop-down, but changing data in the list does not update any data elsewhere.
- A named range must have a unique name. Because they can be accessed globally, the spreadsheet needs to have distinct names for the named ranges.
- A named range's name can't contain a space. In a spreadsheet, a space denotes a new term, such as a new variable or new sheet name. You can use an underscore instead of a space in the name of a named range, or you can use camel case (for example, `Monster_Names`, or `MonsterNames`).
- Named ranges are not case sensitive, so `CargoWeight`, `Cargoweight`, and `cargoweight` are all treated as exactly the same name.

Further Steps

After completing this chapter, you should take some time to practice in the real world with the concepts covered here. Try these exercises to further investigate spreadsheet basics:

- The best way to learn to use a spreadsheet is to use spreadsheets a lot! Start thinking about ways that a spreadsheet might be useful. Making a budget is a common use, but there are also many more. You could, for example, use a spreadsheet to make an inventory of a home or an office. Or you could use a spreadsheet to track exercise numbers, weight change over time, or other health-related data. Spreadsheets are great for organizing catalogs of music, movies, and games.
- Spreadsheets can and should be used for game development. To get an idea of how helpful they are, find some game data for a game you like. Most popular games have fan sites or even official sites that show some form of data that you can copy and paste into a spreadsheet and then manipulate. By practicing with data that others have created, you can see some best practices in action and possibly some pitfalls to avoid when you start making spreadsheets with your own data.

SPREADSHEET FUNCTIONS

Spreadsheets can calculate more than simple formulas; they can use *functions*, which are prewritten bits of code, each of which does a specific task. End users, including game system designers, use functions to quickly perform complex tasks without needing to know exactly what the underlying code does. Every function has a function name and parameters. The function is what should happen, and the parameters accept bits of data called *arguments* and tell the function what data to use as it performs the function.

Grouping Arguments

Computer programs do not know English intuitively as humans do, so you must write functions and input data in a format that the computer can understand without any ambiguity.

For example, consider the English language phrase “Pesto butter toast.” Does this phrase mean pesto and butter and toast? Or does it mean “pesto butter” and toast? Or does it mean pesto and “butter toast”? Or does it mean toast that has been coated with a pesto-infused butter mixture? As English speakers, we can use context and custom to figure out the meaning of words, but computers cannot. For a human, they could take vocal cues of emphasis, or visual clues related to what they see in front of them, to figure out the intended meaning. For a computer, you need to spell out what you mean in exacting detail. When inputting data into a function, each piece of data is called an *argument*, and you use commas to separate multiple arguments. The way you group the arguments has meaning for the computer. For example, you could write “pesto butter toast” in several different ways to communicate the exact items you want the spreadsheet to understand. Here are several ways you could break it down for the computer, and how the computer would interpret each one:

- (pesto butter toast) is one item: toast with pesto butter on it.
- (pesto, butter toast) is two items: One is pesto, and the other is buttered toast.
- (pesto butter, toast) is two items: One is mixed pesto butter, and the other is plain toast.
- (pesto, butter, toast) is three items: Each ingredient in the list is separate.

Function Structure

Functions are, in behavior, much like machines in the real world. Things go into a machine, the machine does a preset operation on the things, and the machine returns the things changed in some way. To better understand how functions work, an analogy to using real-world machines can be helpful.

For example, consider a blender, which, as its name suggests, is a machine that blends up food. You, as the user, can place a wide variety of food items in the blender, and then the blender does what it says it does and returns a new product to you. Imagine that the blender is a spreadsheet function. Much like a blender, the function needs a machine, a container, and ingredients in order to do its job. For a spreadsheet, the “machine” is the name of the function, the “container” is a set of parentheses, and the “ingredients” are data, in the form of comma-separated arguments.

As you can see in Figure 6.1, the blender is standing in for the name of the function. It is followed by the container full of ingredients. In this case, multiple ingredients (“any item”) can be placed in the container.

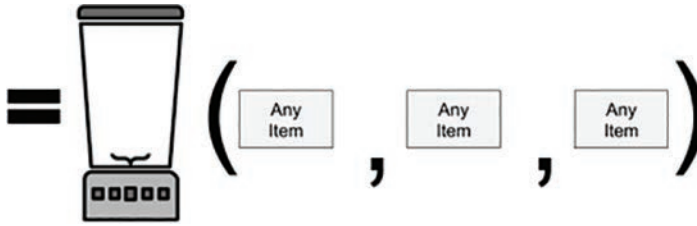


Figure 6.1 A blender as a function



Figure 6.2 A blender with ingredients

Figure 6.2 shows tomato, basil, and garlic placed inside the blender as the data. Once the blender is turned on, it processes these three ingredients to form an output—in this case, tomato sauce. Figure 6.3 shows the same function but with different data.



Figure 6.3 A blender with different ingredients

Again, the blender performs its job and blends together the ice, pineapple, and spirits to form a new output—in this case, a fruity drink. This is a completely different output from the same function because the data input is different.

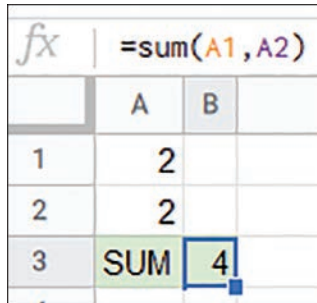
We can break down a spreadsheet function in much the same way we've just broken down the blender ingredients. Let's consider the function SUM. Just as its name suggests, SUM is a function that adds together any data that is placed inside the function container. Here are a few examples:

=SUM(1,1) returns the value 2.

=SUM(1, 1+1, 5, 2) returns the value 10.

Functions allow for formulas or even other functions in their arguments. In the second example above, for instance, SUM would add 1+1 in the argument first, and then it would add the other arguments to get a total of 10.

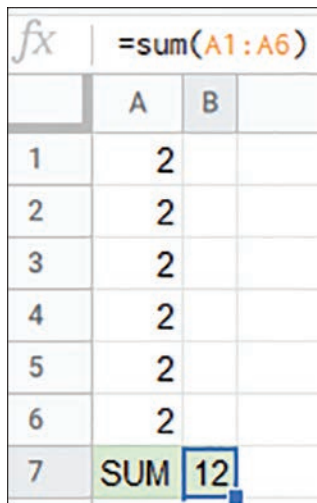
In the example in Figure 6.4, SUM is adding up the contents of two cells. Each cell is referred to in the arguments by the cell address.



	A	B
1	2	
2	2	
3	SUM	4

Figure 6.4 A sum or a reference

In the example in Figure 6.5, SUM is adding the contents of a range that goes from A1 through A6. The colon (:) indicates to the spreadsheet that every cell in the range should be included in this argument. Notice also that different numbers of arguments have been used in this example. Some functions allow the user to input as many or few arguments as they want; SUM is one of them.



	A	B
1	2	
2	2	
3	2	
4	2	
5	2	
6	2	
7	SUM	12

Figure 6.5 The sum of a range