Introduction to Script Language and MATLAB

w. 42. Script Introduction and basic (HG)
- Getting to know Script Language and the MATLAB GUI
- Using variables, vectors and matrices
- Using files (input and output)

w. 43. Programming in Script and Matlab (HG)
- For-loops; while-loops; if-then-else loops

w. 44. Various Matlab Applications (KHH/HG)
- Linear systems, matrix algebra
- Numerical integration and differentiation
- Differential equations

w. 45. Project Work – (KHH + HG)

w. 46. Project Work (KHH + HG)

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming

A SCRIPT IS A SERIES OF INSTRUCTIONS

A script is a series of instructions that a computer can follow to achieve a goal. You could compare scripts to any of the following:

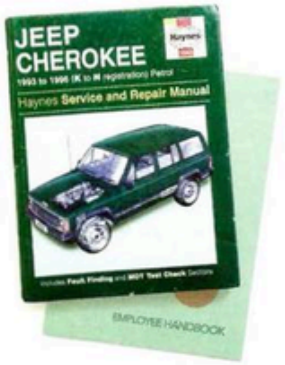
Scripts are made up of instructions a computer can follow step-by-step.

RECIPES

By following the instructions in a recipe, one-by-one in the order set out, cooks can create a dish they have never made before.

Some scripts are simple and only deal with one individual scenario, like a simple recipe for a basic dish. Other scripts can perform many tasks, like a recipe for a complicated three-course meal.

Another similarity is that, if you are new to cooking or programming, there is a lot of new terminology to learn.



HANDBOOKS

Large companies often provide handbooks for new employees that contain procedures to follow in certain situations.

For example, hotel handbooks may contain steps to follow in different scenarios such as when a guest checks in, when a room needs to be tidied, when a fire alarm goes off, and so forth.

In any of these scenarios, the employees need to follow only the steps for that one type of event. (You would not want someone going through every single step in the entire handbook while you were waiting to check in.) Similarly, in a complex script, the browser might use only a subset of the code available at any given time.

MANUALS

Mechanics often refer to car repair manuals when servicing models they are not familiar with. These manuals contain a series of tests to check the key functions of the car are working, along with details of how to fix any issues that arise.

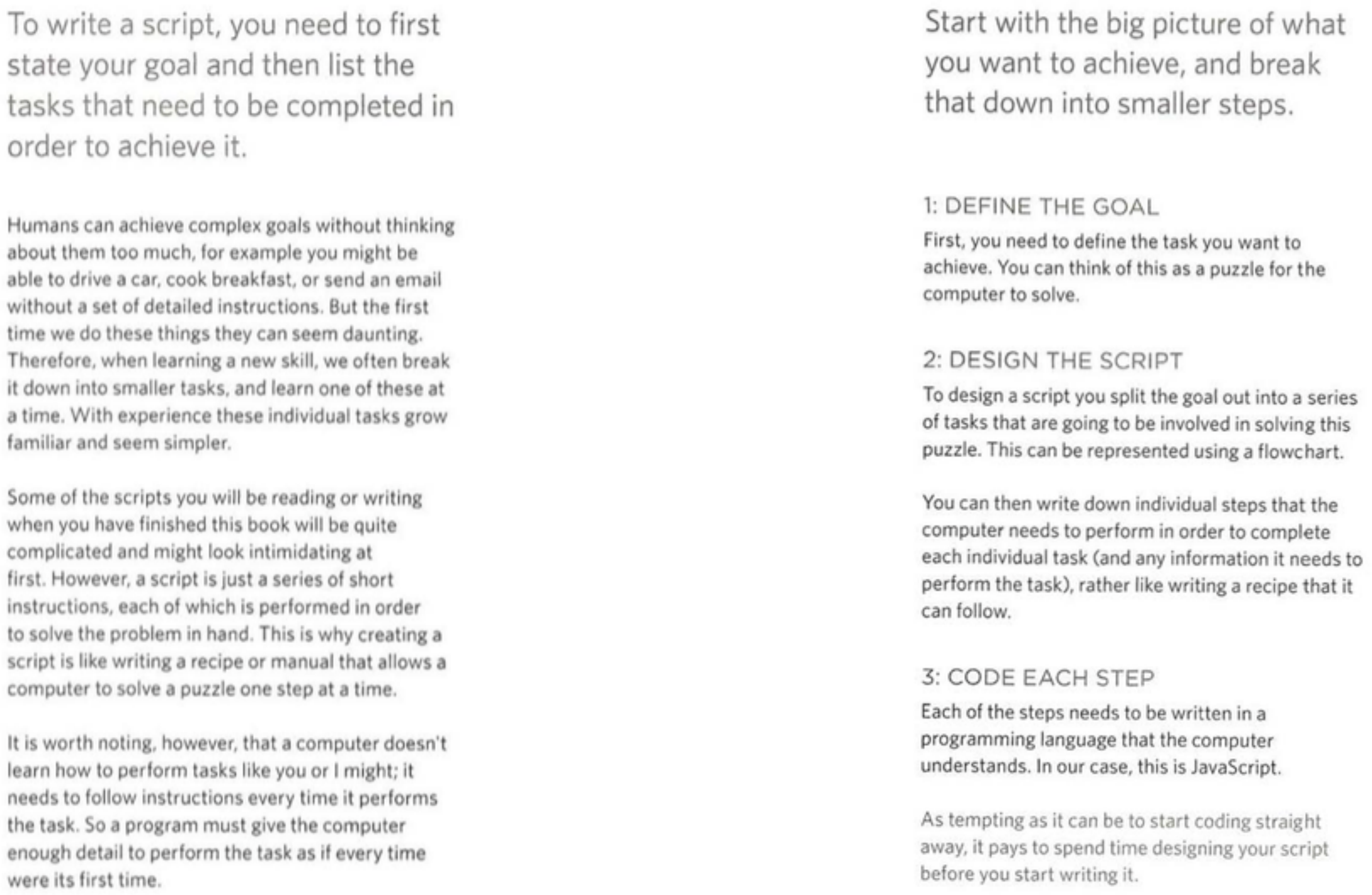
For example, there might be details about how to test the brakes. If they pass this test, the mechanic can then go on to the next test without needing to fix the brakes. But, if they fail, the mechanic will need to follow the instructions to repair them.

The mechanic can then go back and test the brakes again to see if the problem is fixed. If the brakes now pass the test, the mechanic knows they are fixed and can move onto the next test.

Similarly, scripts can allow the browser to check the current situation and only perform a set of steps if that action is appropriate.

from Ducket, J, "Javascript and Jquery, interactive front-end web development", 2014

Introduction to Script Language – ABC of Programming



To write a script, you need to first state your goal and then list the tasks that need to be completed in order to achieve it.

Humans can achieve complex goals without thinking about them too much, for example you might be able to drive a car, cook breakfast, or send an email without a set of detailed instructions. But the first time we do these things they can seem daunting. Therefore, when learning a new skill, we often break it down into smaller tasks, and learn one of these at a time. With experience these individual tasks grow familiar and seem simpler.

Some of the scripts you will be reading or writing when you have finished this book will be quite complicated and might look intimidating at first. However, a script is just a series of short instructions, each of which is performed in order to solve the problem in hand. This is why creating a script is like writing a recipe or manual that allows a computer to solve a puzzle one step at a time.

It is worth noting, however, that a computer doesn't learn how to perform tasks like you or I might; it needs to follow instructions every time it performs the task. So a program must give the computer enough detail to perform the task as if every time were its first time.

Start with the big picture of what you want to achieve, and break that down into smaller steps.

1: DEFINE THE GOAL

First, you need to define the task you want to achieve. You can think of this as a puzzle for the computer to solve.

2: DESIGN THE SCRIPT

To design a script you split the goal out into a series of tasks that are going to be involved in solving this puzzle. This can be represented using a flowchart.

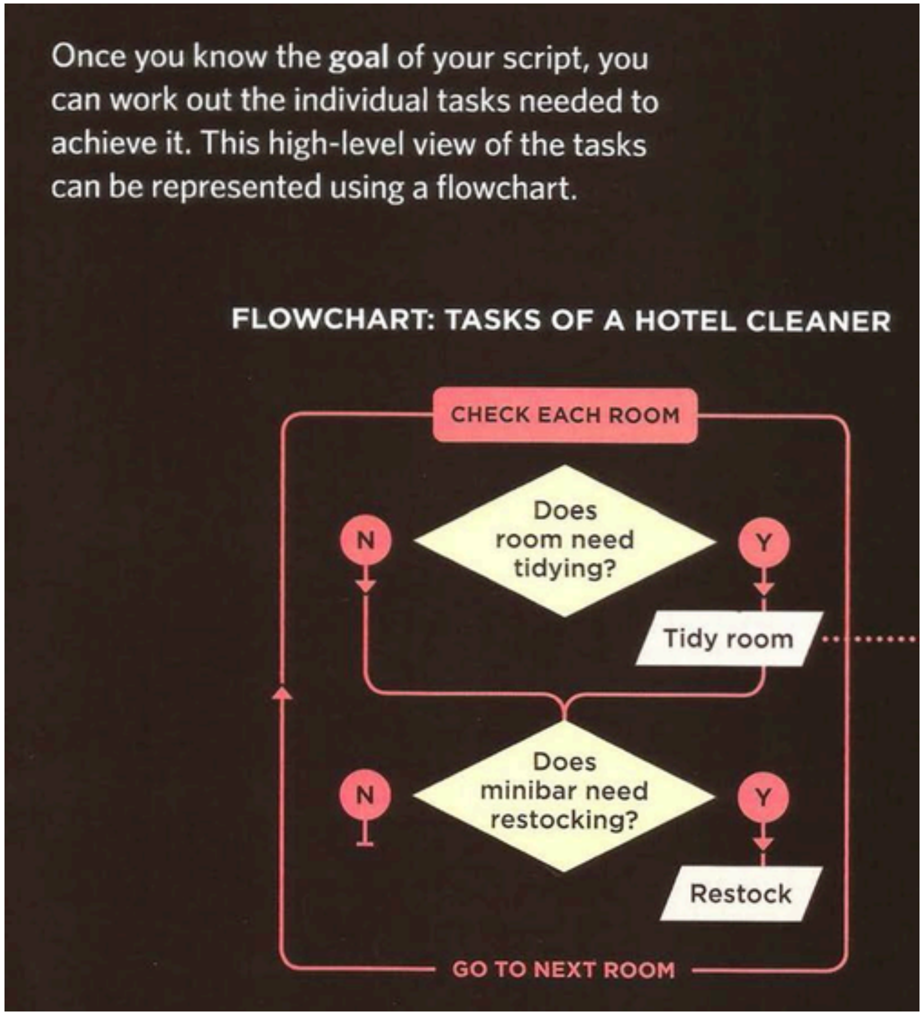
You can then write down individual steps that the computer needs to perform in order to complete each individual task (and any information it needs to perform the task), rather like writing a recipe that it can follow.

3: CODE EACH STEP

Each of the steps needs to be written in a programming language that the computer understands. In our case, this is JavaScript.

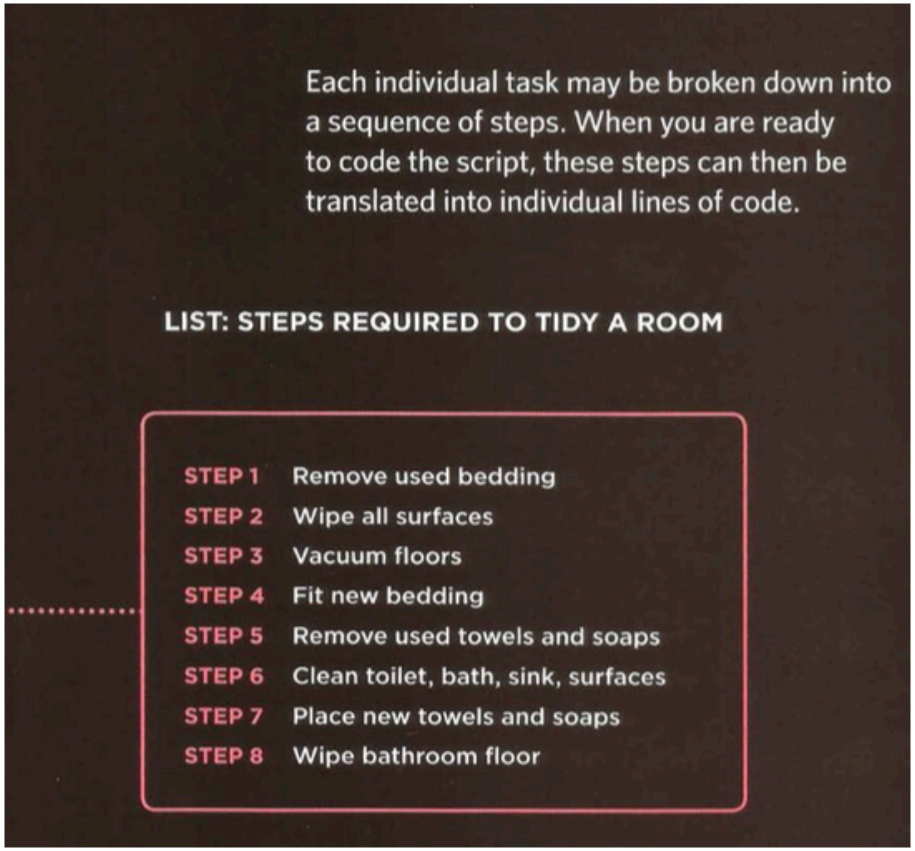
As tempting as it can be to start coding straight away, it pays to spend time designing your script before you start writing it.

Introduction to Script Language – ABC of Programming



Once you know the **goal** of your script, you can work out the individual tasks needed to achieve it. This high-level view of the tasks can be represented using a flowchart.

FLOWCHART: TASKS OF A HOTEL CLEANER



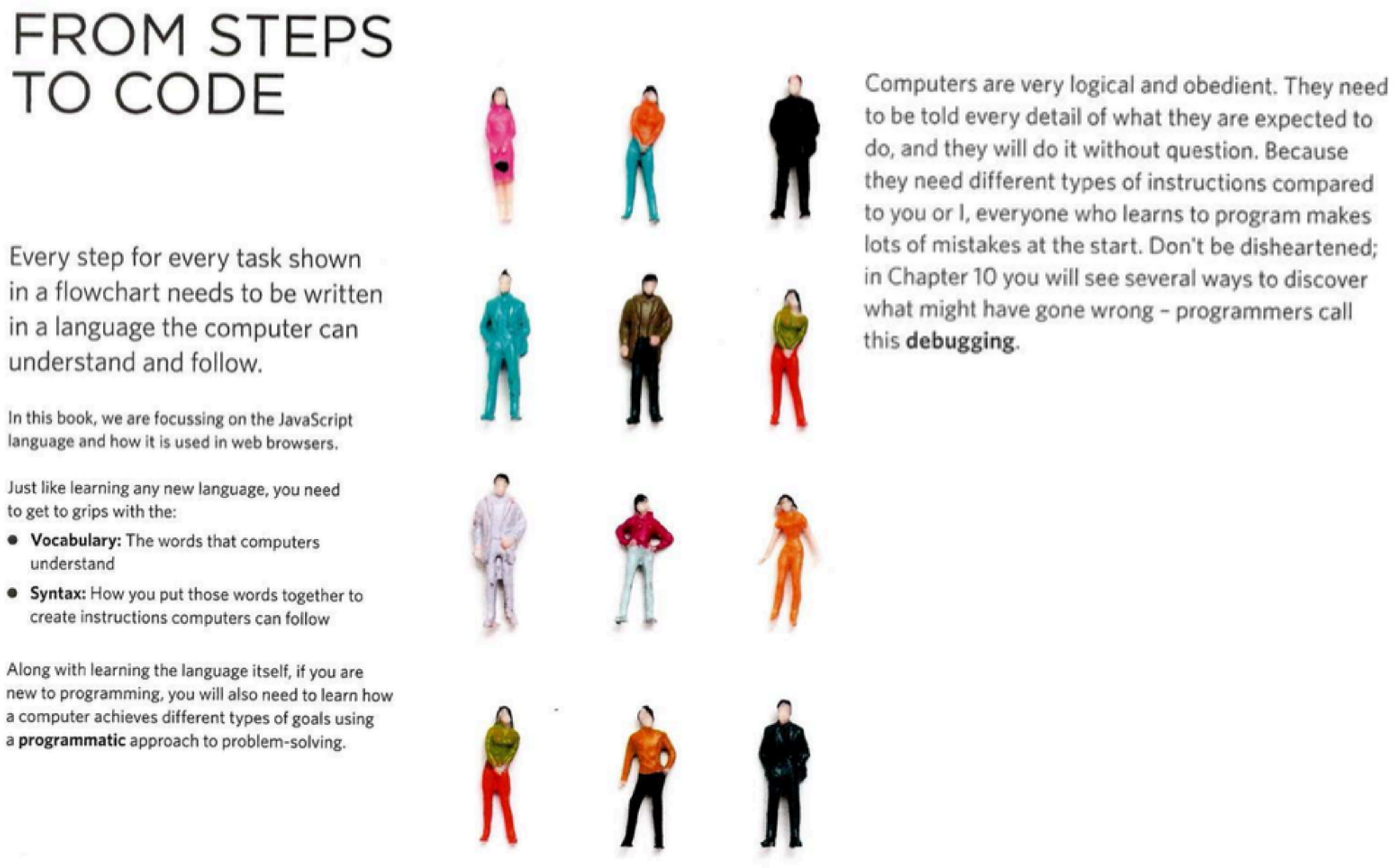
Each individual task may be broken down into a sequence of steps. When you are ready to code the script, these steps can then be translated into individual lines of code.

LIST: STEPS REQUIRED TO TIDY A ROOM

STEP 1	Remove used bedding
STEP 2	Wipe all surfaces
STEP 3	Vacuum floors
STEP 4	Fit new bedding
STEP 5	Remove used towels and soaps
STEP 6	Clean toilet, bath, sink, surfaces
STEP 7	Place new towels and soaps
STEP 8	Wipe bathroom floor

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming



FROM STEPS TO CODE

Every step for every task shown in a flowchart needs to be written in a language the computer can understand and follow.

In this book, we are focussing on the JavaScript language and how it is used in web browsers.

Just like learning any new language, you need to get to grips with the:

- **Vocabulary:** The words that computers understand
- **Syntax:** How you put those words together to create instructions computers can follow

Along with learning the language itself, if you are new to programming, you will also need to learn how a computer achieves different types of goals using a **programmatic** approach to problem-solving.

Computers are very logical and obedient. They need to be told every detail of what they are expected to do, and they will do it without question. Because they need different types of instructions compared to you or I, everyone who learns to program makes lots of mistakes at the start. Don't be disheartened; in Chapter 10 you will see several ways to discover what might have gone wrong - programmers call this **debugging**.

Introduction to Script Language – ABC of Programming

You need to learn to "think" like a computer because they solve tasks in different ways than you or I might approach them.

Computers solve problems **programmatically**; they follow series of instructions, one after another. The type of instructions they need are often different to the type of instructions you might give to another human. Therefore, throughout the book you will not only learn the vocabulary and syntax that JavaScript uses, but you will also learn how to write instructions that computers can follow.

For example, when you look at the picture on the left how do you tell which person is the tallest? A computer would need explicit, step-by-step instructions, such as:

1. Find the height of the first person
2. Assume he or she is the "tallest person"
3. Look at the height of the remaining people one-by-one and compare their height to the "tallest person" you have found so far
4. At each step, if you find someone whose height is greater than the current "tallest person", he or she becomes the new "tallest person"
5. Once you have checked all the people, tell me which one is the tallest

So the computer needs to look at each person in turn, and for each one it performs a test ("Are they taller than the current tallest person?"). Once it has done this for each person it can give its answer.

Consider how you might approach a different type of script. This example calculates the cost of a name plaque. Customers are charged by the letter.

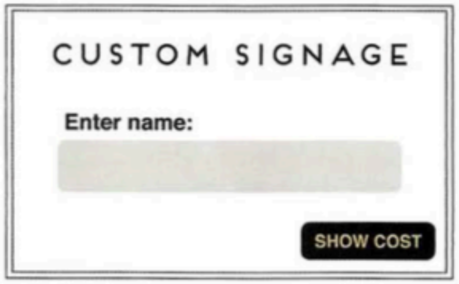
The first thing you should do is detail your goals for the script (what you want it to achieve):

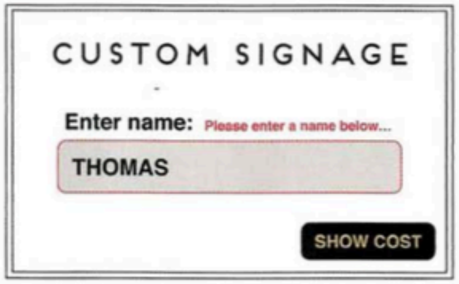
Customers can have a name added to a plaque; each letter costs $5. When a user enters a name, show them how much it will cost.

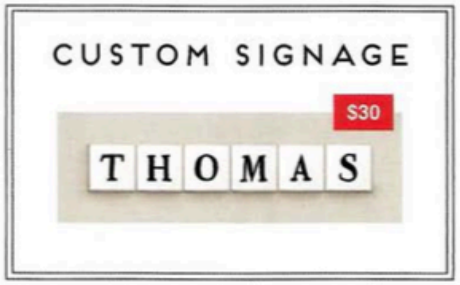
Next, break it into a series of tasks that have to be performed in order to achieve the goals:

1. The script is triggered when the button is clicked.
2. It collects the name entered into the form field.
3. It checks that the user has entered a value.
4. If the user has not entered anything, a message will appear telling them to enter a name.
5. If a name has been entered, calculate the cost of the sign by multiplying the number of letters by the cost per letter.
6. Show how much the plaque costs.

(These numbers correspond with the flowchart on the right-hand page.)







Introduction to Script Language – ABC of Programming

You need to learn to "think" like a computer because they solve tasks in different ways than you or I might approach them.

Computers solve problems **programmatically**; they follow series of instructions, one after another. The type of instructions they need are often different to the type of instructions you might give to another human. Therefore, throughout the book you will not only learn the vocabulary and syntax that JavaScript uses, but you will also learn how to write instructions that computers can follow.

For example, when you look at the picture on the left how do you tell which person is the tallest? A computer would need explicit, step-by-step instructions, such as:

1. Find the height of the first person
2. Assume he or she is the "tallest person"
3. Look at the height of the remaining people one-by-one and compare their height to the "tallest person" you have found so far
4. At each step, if you find someone whose height is greater than the current "tallest person", he or she becomes the new "tallest person"
5. Once you have checked all the people, tell me which one is the tallest

So the computer needs to look at each person in turn, and for each one it performs a test ("Are they taller than the current tallest person?"). Once it has done this for each person it can give its answer.

Consider how you might approach a different type of script. This example calculates the cost of a name plaque. Customers are charged by the letter.

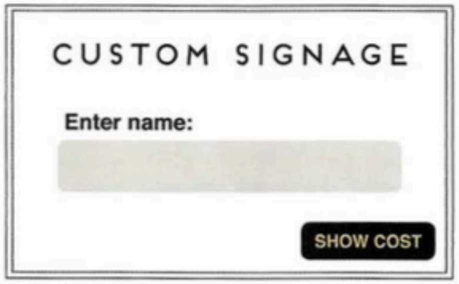
The first thing you should do is detail your goals for the script (what you want it to achieve):

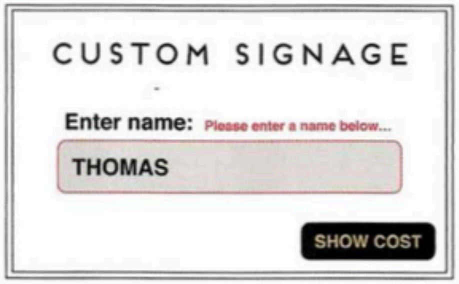
Customers can have a name added to a plaque; each letter costs $5. When a user enters a name, show them how much it will cost.

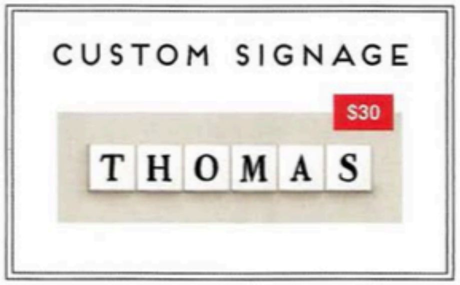
Next, break it into a series of tasks that have to be performed in order to achieve the goals:

1. The script is triggered when the button is clicked.
2. It collects the name entered into the form field.
3. It checks that the user has entered a value.
4. If the user has not entered anything, a message will appear telling them to enter a name.
5. If a name has been entered, calculate the cost of the sign by multiplying the number of letters by the cost per letter.
6. Show how much the plaque costs.

(These numbers correspond with the flowchart on the right-hand page.)

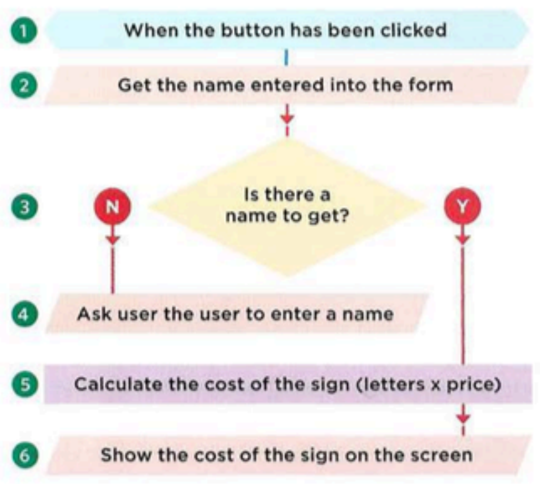






Introduction to Script Language – ABC of Programming

Often scripts will need to perform different tasks in different situations. You can use flowcharts to work out how the tasks fit together. The flowcharts show the paths between each step.

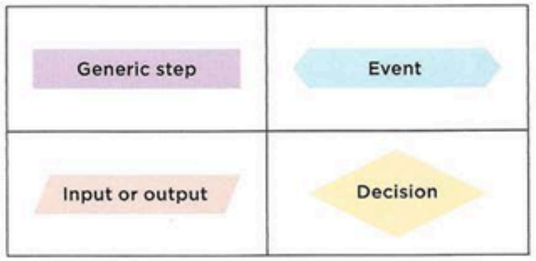

1 When the button has been clicked
2 Get the name entered into the form
3 Is there a name to get?
N
Y
4 Ask user the user to enter a name
5 Calculate the cost of the sign (letters x price)
6 Show the cost of the sign on the screen

Arrows show how the script moves from one task to the next. The different shapes represent different types of tasks. In some places there are decisions which cause the code to follow different paths.

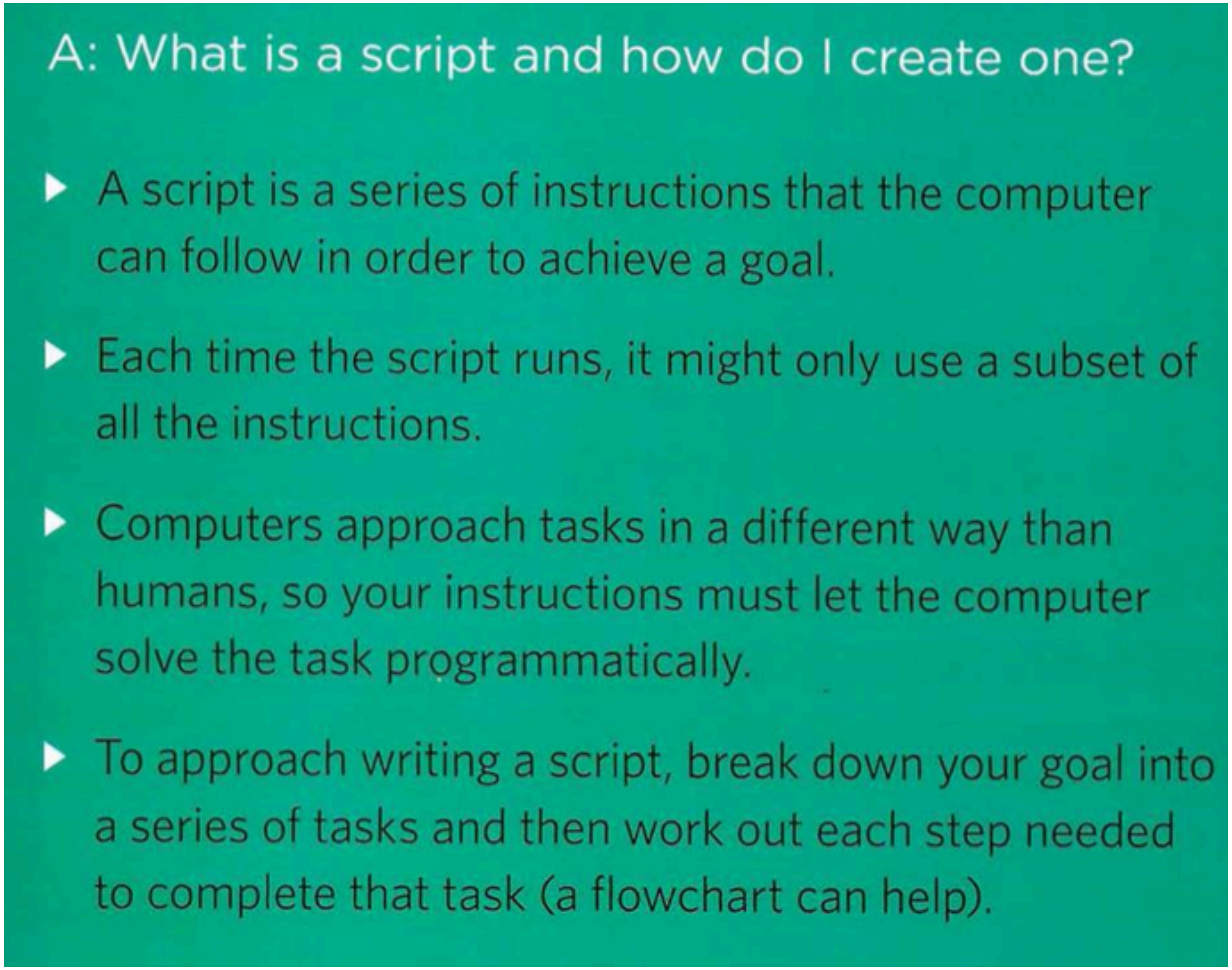
You will learn how to turn this example into code in Chapter 2. You will also see many more examples of different flowcharts throughout the book, and you will meet code that helps you deal with each of these types of situations.

Some experienced programmers use more complex diagram styles that are specifically designed to represent code - however, they have a steeper learning curve. These informal flowcharts will help you understand how scripts work while you are in the process of learning the language.

FLOWCHART KEY



Introduction to Script Language – ABC of Programming



A: What is a script and how do I create one?

- A script is a series of instructions that the computer can follow in order to achieve a goal.
- Each time the script runs, it might only use a subset of all the instructions.
- Computers approach tasks in a different way than humans, so your instructions must let the computer solve the task programmatically.
- To approach writing a script, break down your goal into a series of tasks and then work out each step needed to complete that task (a flowchart can help).

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming

A computer has no predefined concept of what a hotel or car is. It does not know what they are used for. Your laptop or phone will not have a favorite brand of car, nor will it know what star rating your hotel is.

So how do we use computers to create hotel booking apps, or video games where players can race a car? The answer is that programmers create a very different kind of model, especially for computers.

Programmers make these models using data. That is not as strange or as scary as it sounds because the data is all the computer needs in order to follow the instructions you give it to carry out its tasks.

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming

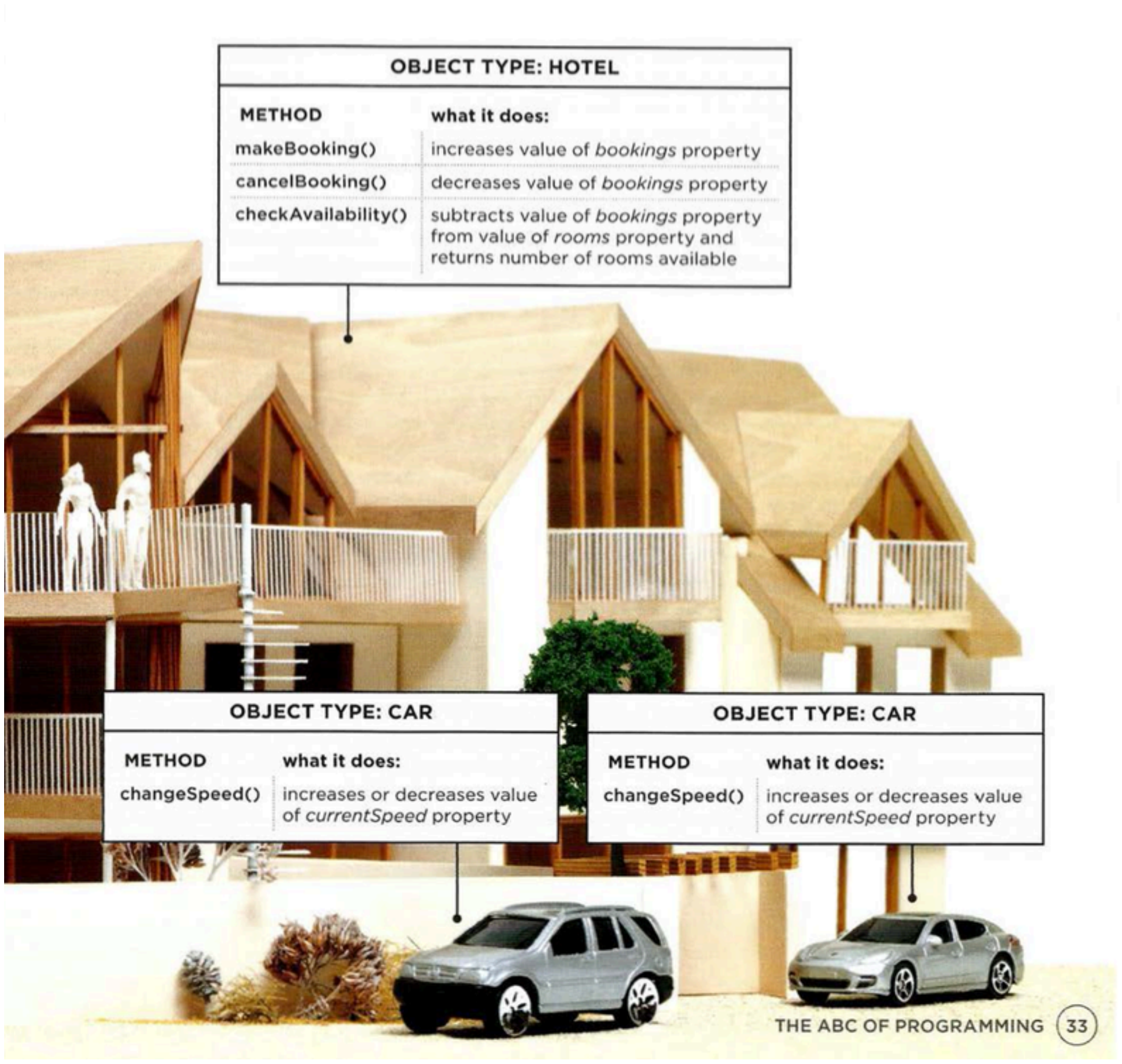
A computer has no predefined concept of what a hotel or car is. It does not know what they are used for. Your laptop or phone will not have a favorite brand of car, nor will it know what star rating your hotel is.

So how do we use computers to create hotel booking apps, or video games where players can race a car? The answer is that programmers create a very different kind of model, especially for computers.

Programmers make these models using data. That is not as strange or as scary as it sounds because the data is all the computer needs in order to follow the instructions you give it to carry out its tasks.

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming



OBJECT TYPE: HOTEL

METHOD	what it does:
makeBooking()	increases value of *bookings* property
cancelBooking()	decreases value of *bookings* property
checkAvailability()	subtracts value of *bookings* property from value of *rooms* property and returns number of rooms available

OBJECT TYPE: CAR

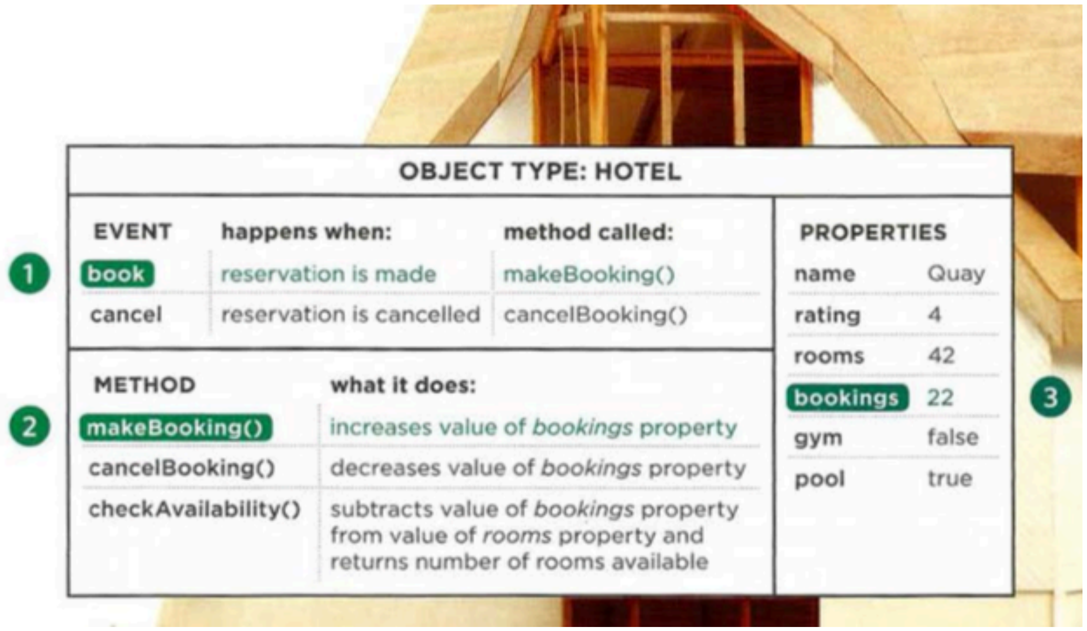
METHOD	what it does:
changeSpeed()	increases or decreases value of *currentSpeed* property

OBJECT TYPE: CAR

METHOD	what it does:
changeSpeed()	increases or decreases value of *currentSpeed* property

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming

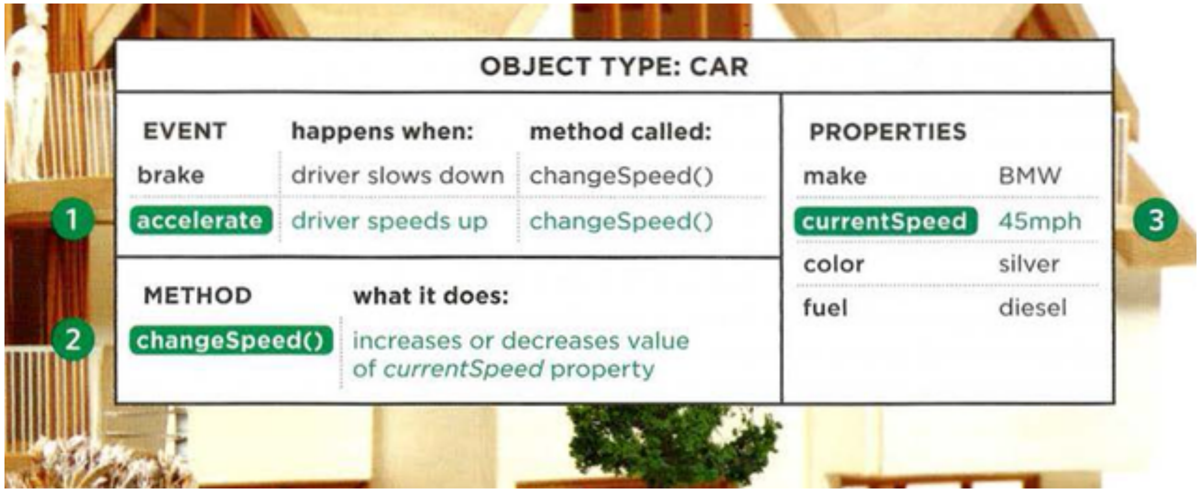


OBJECT TYPE: HOTEL

	EVENT	happens when:	method called:
1	book	reservation is made	makeBooking()
	cancel	reservation is cancelled	cancelBooking()

	METHOD	what it does:
2	makeBooking()	increases value of *bookings* property
	cancelBooking()	decreases value of *bookings* property
	checkAvailability()	subtracts value of *bookings* property from value of *rooms* property and returns number of rooms available

PROPERTIES		
name	Quay	
rating	4	
rooms	42	
bookings	22	3
gym	false	
pool	true	



OBJECT TYPE: CAR

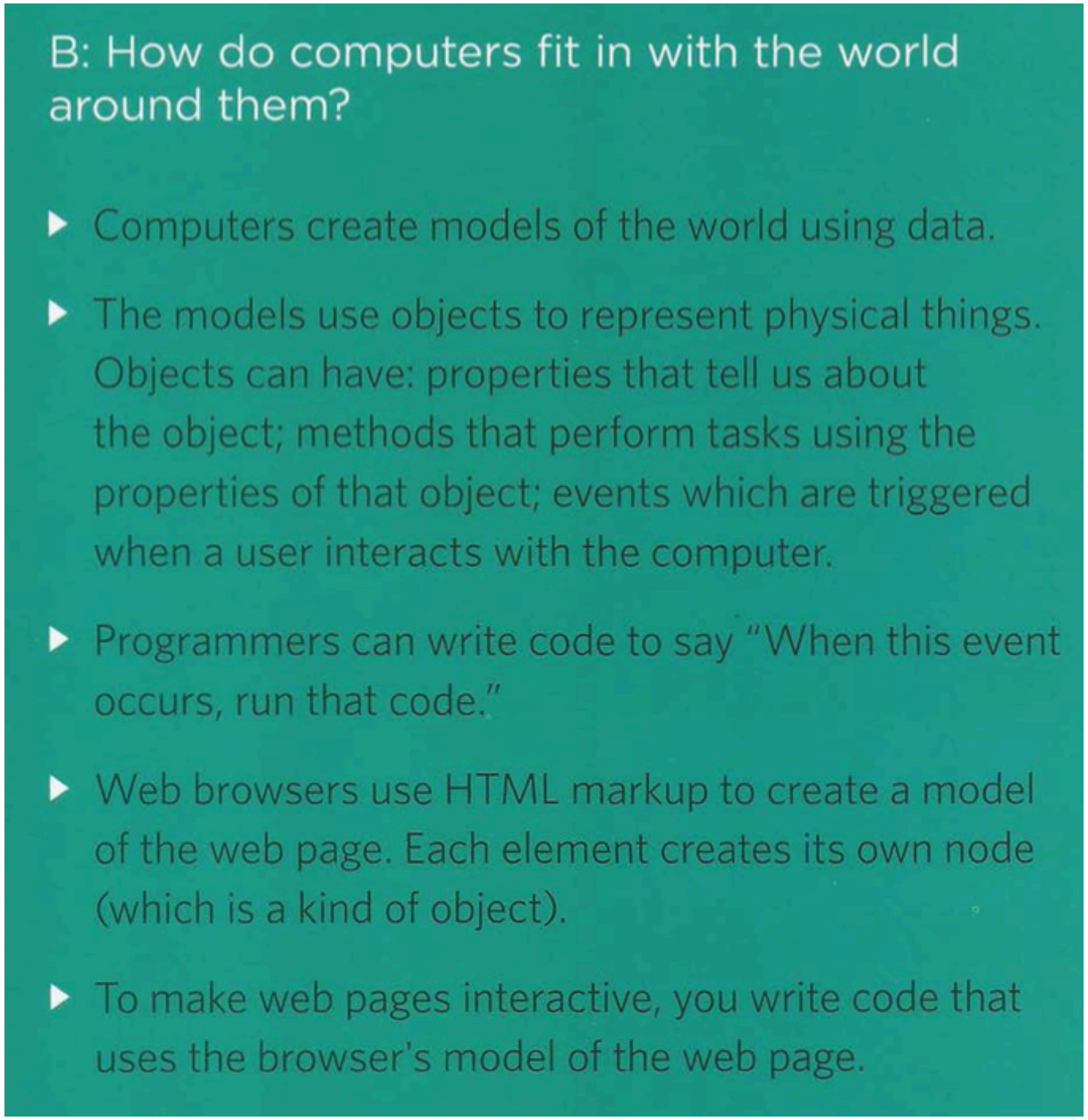
	EVENT	happens when:	method called:
	brake	driver slows down	changeSpeed()
1	accelerate	driver speeds up	changeSpeed()

	METHOD	what it does:
2	changeSpeed()	increases or decreases value of *currentSpeed* property

PROPERTIES		
make	BMW	
currentSpeed	45mph	3
color	silver	
fuel	diesel	

HØGSKOLEN I ÅLESUND

Introduction to Script Language – ABC of Programming



B: How do computers fit in with the world around them?

- Computers create models of the world using data.
- The models use objects to represent physical things. Objects can have: properties that tell us about the object; methods that perform tasks using the properties of that object; events which are triggered when a user interacts with the computer.
- Programmers can write code to say "When this event occurs, run that code."
- Web browsers use HTML markup to create a model of the web page. Each element creates its own node (which is a kind of object).
- To make web pages interactive, you write code that uses the browser's model of the web page.

from Ducket, J, "Javascript and Jquery, interactive front-end web development", 2014

HØGSKOLEN I ÅLESUND