Object Oriented Programming in MATLAB

Greg Reese, Ph.D
Research Computing Support Group
Academic Technology Services
Miami University

October 2013
Object Oriented Programming in MATLAB

© 2013 Greg Reese. All rights reserved
The Problem

Q: What is the problem with writing large software programs?
A: They are very complex. As a result, they tend to be, or rather, are almost always
• Over time
• Over budget
• Bug ridden
• Brittle
  – Making a small change in the system causes many and widespread problems
The Problem

Can sum up problem this way

IF CITIES WERE BUILT LIKE SOFTWARE, THE FIRST WOODPECKER TO COME ALONG WOULD LEVEL CIVILIZATION
The Problem

Object Oriented Programming

- One approach to managing complexity in software and building high-quality programs
- Tends to produce products that are
  - More reliable
  - Extensible
  - Robust, i.e., not brittle
  - Delivered on time and on budget
  - Meet customers’ expectations and requirements
The Problem

Object oriented programming is being widely used because it works!

Object oriented programming is the best way we currently know of building large, complex software systems.
What is Object Oriented Programming (OOP)?

It is a way of designing and developing software.

Its principal characteristic is to make and use (software) models of things in your problem

Models hide detail and make things easier to understand
Object Oriented Programming

Procedural programming
- Write software solution as data and procedures
- Data stored in various forms (structures)
- Procedures, e.g., subroutines, functions, take data in, process it, spit out results

In procedural programming
- Pass data to first procedure, which computes one kind of output
- Pass data to second procedure, which computes another kind of output
- etc.
Object Oriented Programming

IMPORTANT CHARACTERISTIC OF PROCEDURAL PROGRAMMING

Data and procedures are separate!

Most MATLAB programming is procedural
Object oriented programming

• Procedures and data on which they operate are put together in one bundle – an object

• Program “asks” object to “perform a service”
  – Perform a service: object does some operation on the data stored inside itself and returns result
  – Program “asks” because it does not itself send data to procedures. It doesn’t know how object performs service. Program just wants object to give it the result, regardless of how object gets that result
  – “Asking an object to do something” often phrased as “Sending the object a message to do something”
Object Oriented Programming

IMPORTANT CHARACTERISTIC OF OBJECT ORIENTED PROGRAMMING

Data and procedures are bundled together!

MATLAB can do object oriented programming

I find that OOP in MATLAB

• Is easier than working with complex structs
• Makes programs cleaner, i.e., simpler and easier to read
• Is more enjoyable than procedural programming
Object Oriented Programming

Some goals of OOP are to increase

• Abstraction
  – Provide a simplified view of something the user is interested in
  – Provide all features and services user wants and nothing else

• Encapsulation
  – Keep details of items hidden within models

• Comprehension
  – Make software code easier to understand
Object Oriented Programming

Some goals of OOP are to make software more

- **Correct**
  - Meet requirements

- **Robust**
  - Tolerate unexpected use well

- **Reusable**
  - Use same code in different programs

- **Extendable**
  - Add new functionality

- **Maintainable**
  - Fix bugs
OOP Design Process

Analyze domain
• Services?
• Data?
• Simplify by abstracting

Design
• Specify classes
• Specify interfaces

Write code

Test/use code
• Create/run objects
Domain Analysis

*Domain analysis* – studying and understanding the problem and its context in order to make correct models

- Domain is often technical and/or specialized, e.g., crystallography, spectroscopy, stock derivatives, plant growth
  - Usually need a *domain expert*, i.e., someone who specializes in the field
  - Programmer seldom knows domain. Domain expert seldom knows programming!
Domain Analysis

In domain analysis (for models)

• List things that are required, e.g.,
  – What data?
  – What services (actions)?
  – Who will use the models?

• Often helps to list things that are not interested in
  – Affects design of software
  – Makes it clear to customer and programmer that there are certain things the software will not do
Domain Analysis

**WORKING EXAMPLE**

*Model of a snack vending-machine*

**NOTE** – typical OOP sequence is

- All analysis
- All design
- All classes
- All usage
Domain Analysis

Because we’re just learning all parts of OOP, we’ll take it step by step

Analyze service 1 → Design service 1 → Write service 1 → Use service 1

Analyze service 2 → Design service 2 → Write service 2 → Use service 2

etc.
Analysis

**WORKING EXAMPLE**

*Model of a snack vending-machine*

What is context?

Possible question – How is user of model related or connected to machine?

- Buyer of snacks
- Machine manufacturer
- Machine installer
- Machine maintenance

In our example, only look at things from a snack buyer’s point of view
Analysis

What would a buyer care about?

– Imagine going to a snack machine. What do you look at?

- Cost of snacks?
- Selection?
- Machine working?
- Machine clean?
- Cash?
- Muula?
- Gives change?

- Takes bills?
- Credit cards?
- Nutrition information?
- Degradable packaging?
- Need exact change?
Analysis

What would a buyer not care about?

- Imagine going to a snack machine. What do you not care about?

- Machine manufacturer
- Electrical consumption
- Color
- Internal mechanisms
Analysis

Even in things we do care about, there are details that we don’t care about. For example, even if it’s important to know if the machine takes cash, we may not care about, or care to model

– If it accepts foreign currency
– If it takes bills
– What denomination of bills it takes
– If it accepts coins
– If it gives change
– What happens if a coin jams
Analysis

WORKING EXAMPLE

For brevity, will only have four services that our snack machine must provide the user (rest of the program). It must be able to

1. Tell us if it is working
2. Tell us if it accepts cash
3. Tell us the names of all its snacks, their costs, and their availabilities
4. Accept our money and give us our snack
Requirement 1 – Is it working? Suppose machine manufacturer says that if machine is plugged in it will work and (obviously) if it’s not plugged in it won’t work. Therefore, let’s require a service that tells us if plugged in (“yes”) or not plugged in (“no”).

– Service might not literally return words “yes” and “no”, but have some binary equivalent, e.g., 1/0, true/false. Will decide in design phase.
Design

Requirement 1 – Is it working?
Require a service that tells us if plugged in or not. How do this?

Various ways in MATLAB
• Return ‘Y’ or ‘N’
• Return ‘Yes’ or ‘No’
• Return 1 or 0
• Return true or false
Because only need binary response, and MATLAB logical values are inherently binary (keywords `true` and `false`), will choose that way of answering question

- Avoids capitalization issues with “Y”, “N”, “Yes”, “No”
Design

ASIDE

Two common ways in MATLAB procedural programming to get some value are
1. Have a function compute it
2. Get it from a variable

Example

```matlab
>> testResult = isWaterDirty( lead, arsenic );
>> ...
>> isDirty = isWaterDirty( lead, arsenic );
or
>> isDirty = testResult;
```
Design

ASIDE

Only two ways§ to get some value from MATLAB object

1. Have some function in object return it
2. Have some variable in object provide it

Variable

– Pro: shorter, i.e., less typing (as in previous slide)
– Con: if need to give object information so it can compute value, can’t do so

Function – opposite pro/con

§ Common ways, can get value indirectly through unusual ways, e.g., global variables
Requirement 1 – Is it working?

Because don’t need to give object data in order for it to know if it’s plugged in, will go with simpler method, i.e., data in object
Design

MATLAB Terminology

class members – data or functions that are part of a class

property – data that is a member of a class

method – function that is a member of a class
Design

Requirement 1 – Is it working?

Design – class will provide

- **property called** `IsPluggedIn`
  - scalar
  - data type is logical
    - `true` if object plugged in
    - `false` if object not plugged in
Coding

To define a class called, for example, MyClass

1. Make a file called MyClass.m to put the class (and nothing else) in

2. First line of code must be
   \texttt{classdef MyClass \textless handle}

3. Last line of code must be corresponding end statement
   \texttt{end}

Will discuss this later
Try It

Let’s call our class SnackMachine, so create a file called SnackMachine.m with the lines

classdef SnackMachine < handle
end  % end of  classdef...
The file is going to have a lot of “end” statements. Document what an “end” corresponds to

```
classdef SnackMachine < handle
end  % end of classdef...
```
To define a class property, add a section within the class definition that starts with
properties
lists the properties by name, e.g.,
Prop1;
Prop2;
and ends with
end
Better yet, add a comment, e.g.,
end % end of properties
classdef SnackMachine < handle

properties
  IsPluggedIn;
end  % end of properties

end  % end of classdef...
Testing and Using

Terminology – Class vs. Object

Class – template from which to build a piece of software

- Does not exist in executable program
- Directions to build something, but not that thing itself

Object – piece of software built according to a class

- Exists in executable program (memory)
- Built according to directions in a class
Analogy - Class/Object : Blueprint/House

Blueprint – plan for building a house, not the house itself

Class  Blueprint  House – a thing built by plan in a blueprint

Object  House
Testing and Using

Terminology – Class vs. Object

Distinction between class and object

• Often not relevant
• Sometimes though, very important
  – In these cases, will explicitly point out differences
To create an object from a class, make a variable name, followed by the equals sign, followed by the class name, e.g.,

```python
>> myCar = Car;
```

Accessing a property of an object

- Same as accessing field of `struct`
- Use the object name (not the class name), followed by a period and the property name, e.g.,

```python
>> cost = myCar.PurchasePrice;
>> myCar.ModelYear = 2013;
```
To test our class, let's make a function called testSnackMachine and store it in testSnackMachine.m

First we'll just display value of property

```matlab
function testSnackMachine
    snackVendor = SnackMachine;
    % display value of property
    pluggedIn = snackVendor.IsPluggedIn
    whos pluggedIn
```
Coding

Try It

>> testSnackMachine
pluggedIn = []

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluggedIn</td>
<td>0x0</td>
<td>0</td>
<td>double</td>
</tr>
</tbody>
</table>

Two problems already – 1) data type is double, not logical; 2) value is empty array, not true or false

Lesson – MATLAB sets uninitialized class properties to the empty array

• Must always remember to initialize properties
Testing and Using

Two ways to initialize a property

1. In property definition, set value, like assigning value to a variable
   - When MATLAB creates object, gives it that value, but never automatically gives it a value after that

Assume that our snack machine is plugged in, so initialize `IsPluggedIn` to `true`
classdef SnackMachine < handle
properties
    IsPluggedIn = true;
end  % end of properties
end  % end of classdef...
Coding

Try It

Test

```
>> testSnackMachine
pluggedIn = 1  % MATLAB displays true/false as 1/0

Name     Size  Bytes  Class
pluggedIn 1x1    1    logical
```

Much better!
Second way to initialize a property

Because it’s so important to initialize properties and because it’s easy to forget to do so, MATLAB provides a convenient way to initialize an object - the constructor
A constructor is a special method (function) of a class that MATLAB runs whenever you create an object of that class.

The chief function of the constructor is to return a fully usable object.
Coding

Use constructor to
• Initialize properties
• Perform start-up computations
• Acquire resources such as CPUs and files, etc.

Once the constructor finishes, the object must be completely valid, i.e., all methods and properties functioning as defined
Some rules of MATLAB constructors

- Define it in a methods section
- Name must be exactly the same as class name
- Must return one and only one output argument – the constructed object
- Must never return an empty object
Coding

Except in cases to be discussed much later, you don’t need to provide a constructor for your class. If you don’t, MATLAB supplies a constructor that

• Takes no arguments
• Returns a scalar object (as opposed to an array of objects) whose properties are
  – Set to the values specified in the property definitions
  – Set to empty if no values are specified in definitions
You can call the output argument whatever you want, but when referring to the object’s properties or methods from within the constructor, you must use the output-argument name. For example,
If MyClass has a property called Property1 and a method called loadData, the constructor would access them as

```plaintext
function obj = MyClass
    obj.Property1 = 49;
    obj.loadData( 'myFile.txt' );
end
```

Coding

For now, will concentrate on constructors with no arguments

**Try It**

Remove value from property definition, write constructor and in it set property value, and test
classdef SnackMachine < handle

properties
    IsPluggedIn;
end % end of properties

methods

    function obj = SnackMachine
        obj.IsPluggedIn = true;
    end
end % end of methods

end % end of classdef...
Try It

Test

>> testSnackMachine

pluggedIn = 1

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluggedIn</td>
<td>1x1</td>
<td>1</td>
<td>logical</td>
</tr>
</tbody>
</table>

Even though property value not set in definition, it is set in constructor, so result is good
Coding

Q: Initialize properties at definition or in constructor?

A: Either or both. It’s a matter of style

At definition

– Easier to spot uninitialized property
– Initializations may be scattered throughout many properties sections

In constructor

– If have many properties, easy to forget to initialize one
– Good to have all initialization in one spot
Coding

Scenario

• User wants to unplug machine
• Tells it that but uses ‘N’ (for “no”) instead of `false`
• Verifies that not plugged in
Coding

function testSnackMachine
    snackVendor = SnackMachine;
    pluggedIn = snackVendor.IsPluggedIn

    % unplug it
    snackVendor.IsPluggedIn = 'N';

    % verify unplugged
    if snackVendor.IsPluggedIn
        disp( 'Still plugged in' );
    else
        disp( 'Not plugged in' );
    end
Coding

Test
>> testSnackMachine
pluggedIn = 1
Still plugged in
????

Problem is line
if snackVendor.IsPluggedIn

MATLAB
• Evaluates snackVendor.IsPluggedIn to get ‘N’
• ‘N’ is a character whose numerical (ASCII) value is 78
• 78 is nonzero, which is true
Coding

Big problem – user code expects `IsPluggedIn` to be true or false only. Not correct expectation

Bigger problem – all code inside `SnackMachine` class expects `IsPluggedIn` to be true or false only. Major trouble!
Can we prevent user from setting `IsPluggedIn` to anything but `true` or `false`?

Yes – and will show how later.

More important question – should user even be able to set `IsPluggedIn`?

In our context, no! User there to buy junk food. Not his job to tell machine if it’s plugged in or not. Machine should handle that itself.
Coding

Want user to be able to read or get IsPluggedIn but not write to or set IsPluggedIn. Can we do this?

Yes. Can modify behavior of properties by setting their attributes

In MATLAB help system, search for “property attributes” to see list of attributes and their possible values
To set attributes of properties, after “properties” keyword and within parentheses, list attributes and their desired values like this

```
properties(Attribute1=value1,Attribute2=value2)
  Property1;
  Property2;
  Property3; % etc
end
```
Coding

• All properties in a properties section have the same attributes
• You must make a separate properties section for each different combination of attributes that you want
• Any attributes not listed in properties section take on their default values, as defined in the MATLAB documentation
Coding

To control reading from and writing to a property

• To allow all class members and all users to read a property (get its value) set the attribute “GetAccess” to “public”

• To allow only class members to read a property set “GetAccess” to “private”

• To allow all class members and all users to write to a property (set its value) set “SetAccess” to “public”

• To allow only class members to write to a property set “SetAccess” to “private”
Try It
Change your class so that anyone can get `IsPluggedIn` but only class members can set `IsPluggedIn`.

```plaintext
classdef SnackMachine < handle

    properties (GetAccess=public, SetAccess=private)
        IsPluggedIn;
    end % end of properties

    ...% end of classdef...
```
Try It

The default value for both “GetAccess” and “SetAccess” is “public”, so you can also write the change as:

```matlab
classdef SnackMachine < handle
    properties (SetAccess=private)
        IsPluggedIn;
    end % end of properties
end % end of classdef...
```
Try It
Change your test code to read and then write `IsPluggedIn`

```matlab
function testSnackMachine
    snackVendor = SnackMachine;
    snackVendor.IsPluggedIn = false; % write
```

Try It

Run the code

```matlab
>> testSnackMachine
ans = 1 \textcolor{red}{\textbf{Okay to read}}

Setting the 'IsPluggedIn' property of the 'SnackMachine' class is not allowed.

Error in testSnackMachine (line 5)

snackVendor.IsPluggedIn = false; \% write

\textbf{Crash when try to write}
```
Coding

Allowing the user to read `IsPluggedIn` but not to write to it lets him

- Do what he needs to do (find out if the machine is plugged in)
- Prevents him from doing something he doesn’t need to do (change whether or not the machine is plugged in)
  - Which has the benefit of preventing him from storing values that messes up the class and other users
Restricting the type of access we give a user is a special case of a useful software-development guideline called “The Principle of Least Privilege”
The Principle of Least Privilege

Give software only the privileges it needs to work and no more

Here, “privileges” means resources, access rights, security rights, etc.
Coding

Although have fine-tuned `IsPluggedIn`, have lost sight of the bigger picture – the user doesn’t really care if the vending machine is plugged in, he cares if it’s working or not! For example,

- If internal product-delivery mechanism is broken, it’s irrelevant if machine is plugged in or not, it still won’t work
- If machine needs yearly parts maintenance to work and that hasn’t happened, it will shut down whether or not it’s plugged in
Coding

Lesson – make sure you provide the user the service he wants
Try It

The class still needs to get and set `IsPluggedIn` but the user should not be able to do either.

Change the class to effect this and verify that the user can’t read the property
Try It

Change class

classdef SnackMachine < handle

properties(GetAccess=private, SetAccess=private)
    IsPluggedIn;
end % end of properties

... end % end of classdef...
Try It

Run previous test program

function testSnackMachine
snackVendor = SnackMachine;
snackVendor.IsPluggedIn % read
snackVendor.IsPluggedIn = false; % write

Run

>> testSnackMachine

Getting the 'IsPluggedIn' property of the 'SnackMachine' class is not allowed.
Error in testSnackMachine (line 3)

snackVendor.IsPluggedIn % read
Now

• Want user to be able to find out if machine is or isn’t working but not to specify whether that is so

• Want object to be able to find out if machine is or isn’t working and to specify whether it is or isn’t

Let’s add property `IsWorking` to class and let user read but not write to it, and object read and write
Try It

Change class

• Add new properties section because attributes are different than in other section

... properties(GetAccess=public, SetAccess=private)  
    IsWorking;  
end % end of public-get, private-set properties

properties(GetAccess=private, SetAccess=private)  
    IsPluggedIn;  
end % end of private-get, private-set properties

...  

Note added detail in comments of end sections
Coding

Problem – don’t have connection between $\text{IsWorking}$ and $\text{IsPluggedIn}$

By definition, machine is working if and only if it is plugged in, so when code reads value of $\text{IsWorking}$, want it to return value of $\text{IsPluggedIn}$

How do this?
MATLAB does this through dependent properties

*Dependent Property* – a property of a class whose value is computed or derived from other properties

Regular properties are stored in memory. Dependent properties are not stored at all – because they are computed
Concept of a dependent (computed) property is common. Example

Suppose Point is a class that represents a point in the plane. It has two (regular) properties X and Y, representing the obvious coordinates. If user needs to have equivalent polar coordinates, how provide properties for those?
Coding

Option 1 – make two new regular properties, \texttt{Radius} and \texttt{Angle}, to store two polar coordinates

Problem – have now doubled the amount of memory each Point takes

Option 2 – make dependent properties \texttt{Radius} and \texttt{Angle}

\begin{itemize}
  \item When user reads \texttt{Radius}, compute \texttt{sqrt( X\^2 + Y\^2 )} and return it
  \item When user asks for \texttt{Angle}, compute \texttt{atan2( Y, X )} and return it
\end{itemize}
Coding

Code accesses a dependent property the same way it accesses a regular property, e.g.,

```matlab
if myCar.MaxSpeed < 60 \% read access
disp('I need a new car');
end

\% paint my car
myCar.Color = 'red'; \% write access
```
However, when code reads a dependent property, behind the scenes MATLAB replaces the property with a call to the function that computes that dependent property’s value and returns it. That function is called a *property get method*.
Property get method

• Is a class method
• Must be declared in a methods section that does not have any attributes
• Called `get.PropertyName`
  where “PropertyName” is the name of the corresponding property
Example – if the class Car has a dependent property called MilesPerGallon, the property get method would be written as

```matlab
function mpg = get.MilesPerGallon( obj )
    mpg = ... % code to compute output
end
```

- Property name
- Exactly one input
- Must be "get"
- No attributes
- Function must have end statement
- End % end of methods
We’ll study methods more later. For now

• Define a method in a “methods” section
  – Define same way as ordinary function, but MUST have end statement corresponding to function line

• The first input argument comes from MATLAB (not the user)
  – Often called “obj”

• A method in an object has access to all properties in that object
  – Access by appending “.PropertyName” to first method argument, e.g. obj.Gallons
Coding

First step in making dependent property

• Define it in a property section whose “Dependent” attribute is set to “true”
  – Can use other attributes also
  – All properties in this section will be dependent
Try It

Change class

properties (GetAccess=public, ...
SetAccess=private, Dependent=true)

IsWorking;

end % end of public-get, private-set, dependent

...
Coding

Second step is to make the property get method (if you want one)

• Method must go in a methods section that has no attributes

• Method must be called `get.PropertyName`

• Method has exactly one input argument, the one supplied by MATLAB, e.g., “obj”
Try It
Change class
...
methods

function obj = SnackMachine
    obj.IsPluggedIn = true;
end % end of constructor

function working = get.IsWorking( obj )
    working = obj.IsPluggedIn;
end
end
...

Constructor from before
Try It

Change test program to read `IsWorking` function:

```matlab
function testSnackMachine
    snackVendor = SnackMachine;
    snackVendor.IsWorking
```

Run

```
>> testSnackMachine
ans = 1
```
Try It
Change initial value of `IsPluggedIn` and run test program

```matlab
function obj = SnackMachine
    obj.IsPluggedIn = false;
end

..."n
 Run
>> testSnackMachine
ans = 0
Third step is to make the property set method (if you want one). Since we don’t need one for IsWorking, will study later.
Recap
First requirement was to provide service that lets user determine if machine is working or not. Met this requirement by providing two properties of the class, `IsWorking` and `IsPluggedIn`
Design

property IsWorking

• Dependent - yes
• Read access – public
• Write access – none
• Property get method – returns value of IsPluggedIn
• Property set method – none
• Initial value – initial value of IsPluggedIn
Design

property IsPluggedIn

• Dependent - no
• Read access – private
• Write access – private
• Property get method – N/A
• Property set method – N/A
• Initial state – it is plugged in
Design

Whew! All that to answer “Is it working?”, one simple yes/no question.

Since that was so much fun

Let’s Do It Again!
Analysis

Requirement 2 – Does it accept cash?
From before, we saw that there’s a lot more to this question than it first appears. We can ask
- If it accepts foreign currency
- If it takes bills
- What denomination of bills it takes
- If it accepts coins
- If it gives change
- What happens if a coin jams
Analysis

Requirement 2 – Does it accept cash?
We will greatly simplify this by asking for a yes/no type of answer. For now, will always say it accepts cash and answers “yes”
• “yes” means can assume
  – It will take whatever coins or bills we give it
  – It will always be able to give us change
Design

From designing `IsPluggedIn`, have decided that a logical property is good way to answer yes/no question. Let’s make `AcceptsCash`

- Dependent – ?
  - No: assume specified directly by manufacturer
- Read access – ?
  - Public: user needs to get this info
- Write access – ?
  - Private: user shouldn’t be able to set this
- Property get method – N/A
- Property set method – N/A
- Initial state – it does accept cash
Try It

Don’t have a properties section that is public read, private write, not dependent, so make one with `AcceptsCash` there

```plaintext
... properties(GetAccess=public, SetAccess=private)
    AcceptsCash;
end % end of public-get, private-set properties
...```

Coding
Try It

Initialize it in the constructor

... methods

function obj = SnackMachine
    obj.AcceptsCash = true;
    obj.IsPluggedIn = true;
end % end of constructor

...
Try It

Edit test program and run it

```matlab
function testSnackMachine
    snackVendor = SnackMachine;
    if snackVendor.AcceptsCash
        disp('Takes cash!');
    else
        disp('Doesn''t take cash');
    end
end
```

Run

```matlab
>> testSnackMachine
Takes cash!
```
Try It

Change initialization to \texttt{false}

```matlab
function \texttt{obj} = \texttt{SnackMachine}
    \texttt{obj.AcceptsCash} = \texttt{false};
    \texttt{obj.IsPluggedIn} = \texttt{true};
end % end of constructor
```

Run

```matlab
>> \texttt{testSnackMachine}
Doesn't take cash
```

(Change initialization back to \texttt{true})
Recap
Second requirement was to provide service that lets user determine if machine accepts cash or not
Met this requirement with property AcceptsCash

- Dependent - no
- Read access – public
- Write access – private
- Initial state – does accept cash
Analysis

Requirement 3 was to give us the snack selection. Specifically, want
- Names of all snacks (whether in stock or not)
- Costs of all snacks (whether in stock or not)
- Availabilities (in/out of stock) of all snacks

Initialize as follows

<table>
<thead>
<tr>
<th>Snack</th>
<th>Cost</th>
<th>Available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kit Kat</td>
<td>$0.75</td>
<td>Yes</td>
</tr>
<tr>
<td>Coke</td>
<td>$1.50</td>
<td>Yes</td>
</tr>
<tr>
<td>Swedish gummy fish</td>
<td>$1.25</td>
<td>No</td>
</tr>
<tr>
<td>Granola bars</td>
<td>$1.25</td>
<td>Yes</td>
</tr>
<tr>
<td>Monster Energy Drink</td>
<td>$2.25</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Design

Two obvious choices for data type

Structure array

- Pro – easy to use in loops
- Pro – easier for user to move around one array than three
- Con – more complicated than arrays of just one data type

Three arrays

- Pro – easy to use in loops
- Con – must move three arrays instead of one
- Pro – less complicated than struct arrays
Design

Further consideration – a property in MATLAB can only have a single output. A method can return any number of outputs. Thus

• If we want to use a \texttt{struct} array for the output, we can provide it with either a property or a method

• If we want three arrays for the output, we have to provide them through a method

Lesson – technical issues of our programming language can effect design of our class
Design

There doesn’t seem to be a compelling reason to choose one design over the other, so I’ll make an executive decision.

Give the user the snack information in three separate arrays.
IMPORTANT

Note that this defines the class interface, i.e., how the class user interacts with the class. It does **not** require us to store the information inside the class. We will decide that when we get to the coding.
Design

For the internal storage of the snack information, there again doesn’t seem to be a compelling reason to choose one design over the other, so for simplicity let’s store the data in three arrays, which will be the properties Names, Costs, and Availabilities.
Design

For the internal storage of the snack information, there again doesn’t seem to be a compelling reason to choose one design over the other, so for simplicity let’s store the data in three arrays, which will be the properties Names, Costs, and Availabilities
Design

Access?

- Will have method that provides all three pieces of info at once
- User doesn’t have use for individual pieces, e.g., what good is cost of a snack if you don’t know its name or if it’s available

This means all three properties should have private set- and get-access

Dependent?

No, all three will store data, not compute it
Try It  Coding

Put in properties

properties( GetAccess=public, SetAccess=private )
   AcceptsCash;
   Availabilities;
   Costs;
   Names;
end % end of public-get, private-set properties

methods

function obj = SnackMachine
   obj.AcceptsCash = false;
   obj.Availabilities = [true, true, false, true, true, true];
   obj.Costs = [ 0.75, 1.50, 1.25, 1.25, 2.25 ];
   obj.IsPluggedIn = true;
   obj.Names = { 'Kit Kat', 'Coke', 'Swedish gummy fish',...
                 'Granola bars', 'Monster drink' };
end % end of constructor
Design

Method

Need a method that returns, in three separate arrays, information on all snacks.

• Let’s call method “snacks”
• Since the user needs to call it, “Access” must be public
• It returns three arrays
• It has no user-provided inputs
Coding

We’re going to use a method, let’s review and study MATLAB class methods

To define a method, write a function inside a methods section, e.g.,

```matlab
methods
    function obj = myMethod( obj, speed, height )
        ...
    end % end of myMethod
end % end of methods
```
Coding

• Define a method the same way as ordinary function, but it MUST have end statement corresponding to function line

• A method definition always has at least one input argument
  – MATLAB (not the user) passes the first input argument, which is often called “obj”

• Any remaining arguments are those passed by the user (caller)
Let's use the class Car (available in Car.m) for some examples.

There are two ways of using (calling) methods – dot notation and function notation:

- For the most part produce the same results.
- Some exceptions in advanced usage – see MATLAB help for “Dot Notation vs. Function Notation”.
- For our work, will assume both equivalent.
Dot notation

To call a method of an object created by a class, write the object’s name, followed by a dot (period), followed by the method’s name with the appropriate arguments

- Use the object’s name, **not** the class name
- Like accessing a field of a **struct**
Coding

Example

The class Car

- Has a method `addGas` that lets you add gas to the car
  - Method returns the amount of gas actually added, which may be different than passed amount because method won’t let you overflow the tank

- Has two properties
  - `Gas` – how much gas currently in car
  - `MaxGas` – gas-tank size (max gas car can hold)
Coding

Example

methods( Access=public )
...

function gasUsed = addGas( obj, gas )
% INPUTS
% gas - amount of gas to add (gallons)
% OUTPUTS
% gasUsed - amount of gas actually added (gallons)
...

end % end of addGas(...
...

end % end of public-access methods
Example

Call addGas using dot notation

function carDemo

% make car that initially has 10 gallons of gas
myCar = Car( 10 );
fprintf( [ 'My new car comes with %d gallons 
       'of gas but it can hold %d
' ],
       myCar.Gas, myCar.MaxGas );

% fill'er up!
gasToAdd = 50;
gasActuallyAdded = myCar.addGas( gasToAdd );
fprintf( [ 'I tried to add %d gallons but it 
          only took %d\n' ], gasToAdd, gasActuallyAdded );

Run

>> carDemo

My new car comes with 10 gallons of gas but it can hold 25
I tried to add 50 gallons but it only took 15
Function notation

To call a method of an object created by a class, call it as you would a regular function but with the first argument being the object’s name

• Use the object’s name, not the class name
Example

Call `addGas` using dot notation

```matlab
function carDemo

% make car that initially has 10 gallons of gas
myCar = Car( 10 );
fprintf( [ 'My new car comes with %d gallons ...
    'of gas but it can hold %d
' ],...
    myCar.Gas, myCar.MaxGas );

% fill'er up!
gasToAdd = 50;
gasActuallyAdded = addGas( myCar, gasToAdd );
fprintf( [ 'I tried to add %d gallons but it ...
    'only took %d
' ], gasToAdd, gasActuallyAdded );
```

Run

```matlab
>> carDemo
```

My new car comes with 10 gallons of gas but it can hold 25
I tried to add 50 gallons but it only took 15
Dot notation and function notation
• For the most part produce the same results
• Which to use is a matter of preference
  – Preference often based on method in user’s other programming languages
• In these slides, will use dot notation
A method in an object has access to all of that object’s members

• Means both data members (properties) and function members (methods)

• Use properties as if they were regular, local variables, i.e. variables created within the method
The first input argument of a method is always the one that MATLAB inserts, typically called “obj”. Within that method, access other methods and properties of the object by prefixing them with “obj” followed by a period. For example,

- `obj.Property1`
- `obj.method3(speed, height)`
Coding

Example

**addGas method from Car**

```plaintext
function gasUsed = addGas( obj, gas )

% current gas plus amount to add <= full capacity
if obj.Gas + gas <= obj.MaxGas
    obj.Gas = obj.Gas + gas;
    gasUsed = gas;
else
    gasUsed = obj.MaxGas - obj.Gas;
    obj.Gas = obj.MaxGas; % tank filled up
end

end % end of addGas(...)
```

Accessing object properties even though they're not defined in this method.
IMPORTANT

Methods and properties only have access to other methods and properties that are in the same object, not in the same class.

In other words, the methods and properties of one object do not affect the methods and properties of any other object.
function carDemo

% make a Toyota with 10 gallons
toyota = Car( 10 );
% make a Ford with 5 gallons
ford = Car( 5 );
fprintf( 'Toyota has %d gallons\n', toyota.Gas );
fprintf( 'Ford has %d gallons\n\n', ford.Gas );

% add gas to Toyota only
toyota.addGas( 10 );
disp( 'After adding gas to Toyota only...' );
fprintf( 'Toyota has %d gallons\n', toyota.Gas );
fprintf( 'Ford has %d gallons\n\n', ford.Gas );

% add gas to Ford only
disp( 'After adding gas to Ford only...' );
fprintf( 'Toyota has %d gallons\n', toyota.Gas );
fprintf( 'Ford has %d gallons\n', ford.Gas );
Example

``````
>> carDemo
Toyota has 10 gallons
Ford has 5 gallons

After adding gas to Toyota only...
Toyota has 20 gallons
Ford has 5 gallons

call of toyota.addGas(10) affects only Toyota

After adding gas to Ford only...
Toyota has 20 gallons
Ford has 25 gallons

call of ford.addGas(100) affects only Ford
```
Coding

Attributes of methods section

• Will only study one attribute – “Access”
  – Similar to property attributes “GetAccess” and “SetAccess”
  – Only other class members can call methods in a methods-section with attribute “Access=private”
  – Class members and code outside class can call methods in a methods-section with attribute “Access=public”

• All attributes defined in “Table of Method Attributes” in MATLAB help
Try It

Put method in new methods section with public access

methods( Access=public )

function [names, costs, availabilities] = snacks( obj )
    names = obj.NNames;
    costs = obj.CCosts;
    availabilities = obj.AAvailabilities;
end

end % end of public methods
function testSnackMachine
    snackVendor = SnackMachine;
    [ snacks, costs, isAvailable ] = snackVendor.snacks;
    for ii=1:length(snacks)
        fprintf( '%s costs $%.2f Available: %d
', snacks{ii}, costs(ii), isAvailable(ii) );
    end

Run

>> testSnackMachine
Kit Kat costs $0.75 Available: 1
Coke costs $1.50 Available: 1
Swedish gummy fish costs $1.25 Available: 0
Granola bars costs $1.25 Available: 1
Monster drink costs $2.25 Available: 1
Design

Recap for Requirement 3 was to give us the snack selection. Made method called snacks

- Public access
- No user inputs
- Three array outputs, all same size
  - names: string array with snack names
  - costs: array with snack costs (in dollars)
  - availabilities: logical array with snack availability
Analysis

Requirement 4 – Buy snack

Input – snack name and money

Output – snack name and change

Algorithm

• If name matches a snack name and snack is available and enough input money, return snack name and change (difference between input money and snack cost)

• Name comparison will be case-insensitive
Design

Requirement 4 – Buy snack

• Since have inputs and multiple outputs, must be method, not property
• Since user must call method, “Access” is “public”
• Input arguments
  – name: character string
  – money: money in dollars
• Output arguments
  – soldName: input name or empty array (see Algorithm)
  – change: difference between money and cost of snack, or cost of snack itself (see Algorithm)
• Let’s make it buy
Try It

Put method in existing section

methods( Access=public )

function [ soldName, change ] = buy( obj, name, money )
    index = find( strcmpi(name,obj.Names) ); % assume no duplicates
    if ~isempty(index) ...
        && obj.Costs(index)<=money && obj.Availabilities(index)
        soldName = name;
        change = money - obj.Costs(index);
    else
        soldName = [];
        change = money;
    end
end
end
...
end % end of public methods
function testSnackMachine
snackVendor = SnackMachine;
% give yourself $10
money = 10;
% What snacks are there
[ snacks, costs, isAvailable ] = snackVendor.snacks;
% Buy first snack you see
[ name, change ] = snackVendor.buy( snacks{1}, money )
% display resulting situation
if isempty(name)
    fprintf( 'Couldn''t buy %s. Still have $%.2f\n', ... 
               snacks{1}, money );
else
    money = change;
    fprintf( 'Bought %s. Now have $%.2f left\n', ... 
               snacks{1}, money );
end
Try It

Run

>> testSnackMachine
name = Kit Kat
change = 9.2500
Bought Kit Kat. Now have $9.25 left

Insert bogus snack name and run
...

% Buy first snack you see
snacks{1} = 'z';
[ name, change ] = snackVendor.buy( snacks{1}, money )
...

Run

>> testSnackMachine
name = []
change = 10
Couldn't buy z. Still have $10.00
Try It  

Buy first snack you see

function testSnackMachine
    snackVendor = SnackMachine;
    % give yourself $10
    money = 10;
    % What snacks are there
    [ snacks, costs, isAvailable ] = snackVendor.snacks;
    % Buy first snack you see
    [ name, change ] = snackVendor.buy( snacks{1}, money )
    % display resulting situation
    if isempty(name)
        fprintf( 'Couldn''t buy %s. Still have $%.2f\n', ... 
            snacks{1}, money );
    else
        money = change;
        fprintf( 'Bought %s. Now have $%.2f left\n', ... 
            snacks{1}, money );
    end
Object Oriented Programming

Questions?
The End