

**COMS 3101-2 : Programming Languages:  
MATLAB**

# **Object Oriented Programming**



**MATLAB**

**ROHIT SETHI**

# Outline



- Types of programming methodologies
- How to implement OOP in Matlab
- Data and Functions
- Classes : two variations built in Matlab

# Programming Methodologies



Procedural

Functional

Object oriented

# What is Object Oriented Programming



- Class : A **class** is a construct that is used as a blueprint (or template) to create objects of that class.
- Object : specific instance of a class
- Encapsulation: User knows only the functionality of a method not its implementation.
- Inheritance: "Subclasses" are more specialized versions of a class, adding attributes of their own.
- Abstraction: writing code in the most generic class possible.

# Class Defination



Class in Matlab is composed of  
Properties

Methods

That's it ...

# Properties



The image shows a MATLAB Editor window with the following title bar: `C:\Documents\My MATLAB Files\Class examples\VR2008a\Introduc...`. The window contains a MATLAB script defining a class named `sads`. The script is as follows:

```
1  classdef sads
2      % Sensor Array Data Set Class (just properties)
3      properties
4          Wavelength    % Wavelength of sources (m)
5          c=3e8;         % Speed of wave in medium (m/s)
6          NumSensors    % Number of sensors
7          NumSamples    % Number of samples
8          Data           % Sampled sensor data
9          Spacing        % Spacing of array (m)
10         SampleRate    % Sample rate (Hz)
11         Name           % Sensor array test run name
12     end
13 end
```

The status bar at the bottom of the window shows the file name `sads`, the current line number `Ln 11`, the current column number `Col 50`, and the text `OVR`.

# Properties : Access Specifiers



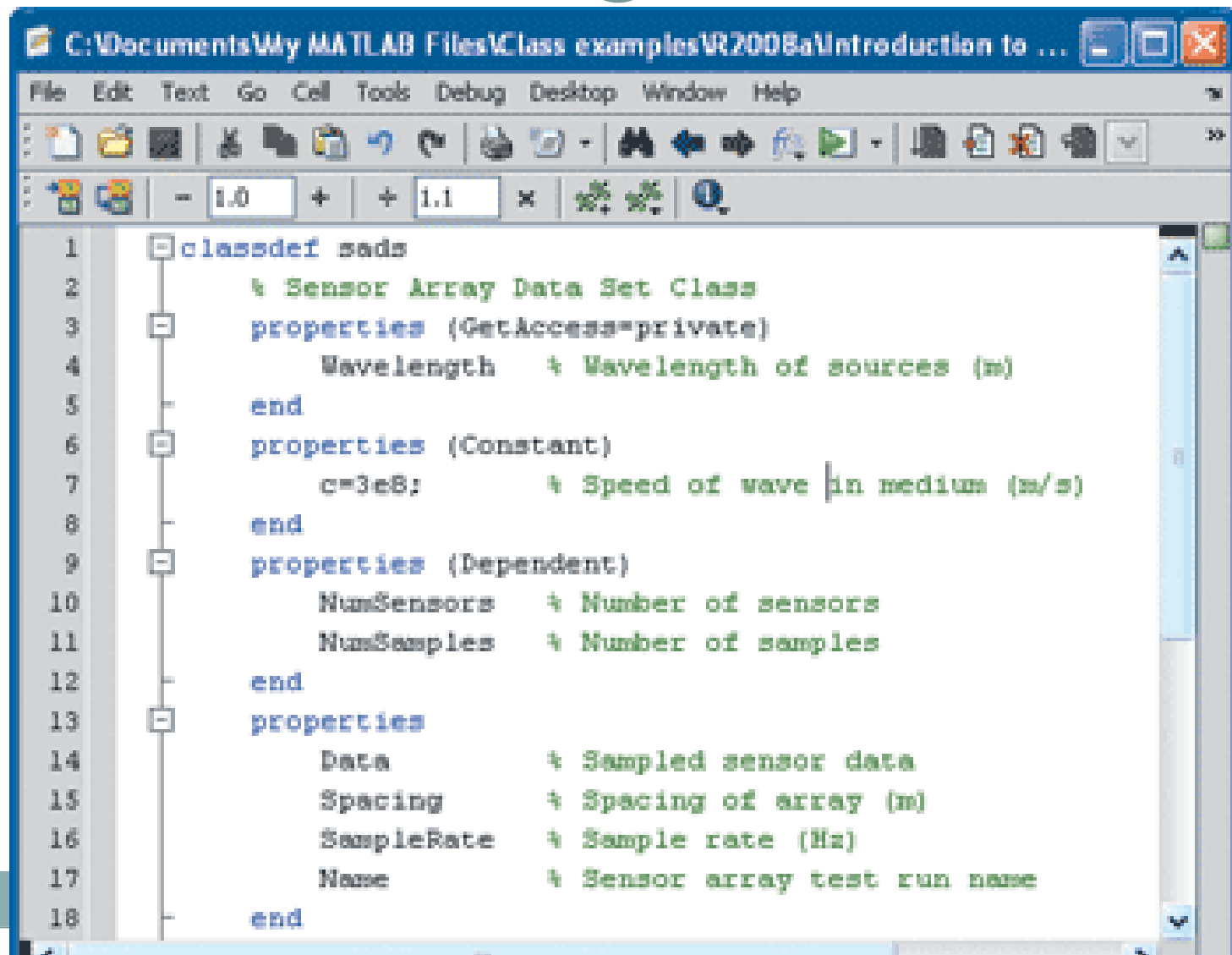
Constant : Value can't be changed.

Can be accessed with class name

SetAccess : Make a property visible only to the methods operating on it by setting the GetAccess attribute to private.

Dependent: Property is calculated only when asked for. Specify a get method that is automatically called when the property is accessed

# Access Control



The image shows a screenshot of the MATLAB Editor window. The title bar indicates the file path: C:\Documents\My MATLAB Files\Class examples\VR2008a\Introduction to ... . The menu bar includes File, Edit, Text, Go, Cell, Tools, Debug, Desktop, Window, and Help. The toolbar contains various icons for file operations, editing, and execution. Below the toolbar, there is a numeric display showing - 1.0 and + 1.1, along with mathematical operators. The main editor area displays the following MATLAB code:

```
1 classdef sads
2     % Sensor Array Data Set Class
3     properties (GetAccess=private)
4         Wavelength % Wavelength of sources (m)
5     end
6     properties (Constant)
7         c=3e8; % Speed of wave in medium (m/s)
8     end
9     properties (Dependent)
10        NumSensors % Number of sensors
11        NumSamples % Number of samples
12    end
13    properties
14        Data % Sampled sensor data
15        Spacing % Spacing of array (m)
16        SampleRate % Sample rate (Hz)
17        Name % Sensor array test run name
18    end
```



# Dependent properties : Get and Set methods



- **Implementation**

```
function NumSensors=get.NumSensors(obj)
    NumSensors=size(obj.Data,2);
end
```

- **Called automatically when properties are accessed**

```
N=s.NumSensors;
```

# Objects and Properties



To Create an object of the above class :

```
>> s=sads;
```

To Set the value of a property

```
>> s.NumSensors=16;
```

# Class Methods

A screenshot of the MATLAB Editor window. The title bar reads "C:\Documents\My MATLAB Files\Class examples\VR2008a\Introduction to OOP\sads.m". The menu bar includes File, Edit, Text, Go, Cell, Tools, Debug, Desktop, Window, and Help. The toolbar contains various icons for file operations, editing, and execution. Below the toolbar, there is a zoom control showing 1.0 and 1.1, and a stack of open files. The main editor area displays the MATLAB code for the 'sads' class. The code is as follows:

```
1  classdef sads
2      % Sensor Array Data Set Class
3      properties (GetAccess=private) ...
6      properties (Constant) ...
9      properties (Dependent) ...
13     properties ...
20     methods
21         function obj=sads(Data, Wavelength, SampleRate, Spacing, Name) ...
32         function plot(obj) ...
50         function [mags, fflip]=magfft(obj, zeroPadTo) ...
71         function magfftplot(obj, zeroPadTo) ...
84         function angles=doa(obj) ...
96         function NumSensors=get.NumSensors(obj) ...
99         function NumSamples=get.NumSamples(obj) ...
102     end
103 end
```

# Example of a method



```
function reset(hObj)
    hObj.m_phase=0;
    reset(hObj.m_H);
    hObj.m_freqTrack = 0;
    hObj.m_diffEncMem = 0;
end
```

- Here hObj is the object of the class defined

# Constructor



```
function obj=sads(Data,  
    Wavelength,SampleRate,Spacing,Name)  
% sads(Data, Wavelength,SampleRate,Spacing,Name)  
  
    obj.Data=Data;  
    obj.SampleRate=SampleRate;  
    obj.Spacing = Spacing;  
    obj.Name=Name;  
    obj.Wavelength=Wavelength;  
end
```

# Destructor



- No need to free memory in classes : Done automatically.
- If any other operations are to be performed, like closing a file, it's implemented in a destructor function named “delete”

```
function delete(obj)
    fclose(obj.FileID);
end
```

# Two Types of Classes



- Value Class : Represent entities that do not need to be unique. The classes we declared were type Value.
- Handle Class : To create a reference to the data contained in an object of the class

Do not want copies of the object to make copies of the object data.

All handle classes are subclasses of the handle class.

# Inheritance



```
classdef employee < handle
    properties
        Name = ''
        Department = '';
    end
    methods
        function e = employee(name,dept)
            e.Name = name;
            e.Department = dept;
        end % employee
        function transfer(obj,newDepartment)
            obj.Department = newDepartment;
        end % transfer
    end
end
```



# Handle Class Behavior



- Handle is an object that references its data indirectly
- creates an object with storage for property values and returns a handle to this object.
- On Assigning the handle to a variable , MATLAB copies just the handle.

# References



- <http://www.mathworks.com/products/matlab/oop-video.html>
- [http://www.mathworks.com/company/newsletters/digest/2008/mar/matlab\\_oop.html](http://www.mathworks.com/company/newsletters/digest/2008/mar/matlab_oop.html)
- [http://www.mathworks.com/access/helpdesk/help/techdoc/matlab\\_oop/brfylvwk-1.html](http://www.mathworks.com/access/helpdesk/help/techdoc/matlab_oop/brfylvwk-1.html)
- [http://www.mathworks.com/access/helpdesk/help/techdoc/matlab\\_oop/exampleindex.html](http://www.mathworks.com/access/helpdesk/help/techdoc/matlab_oop/exampleindex.html)



$Q$

$Q$

$Q$

$Q$