WEC-Sim Training Course

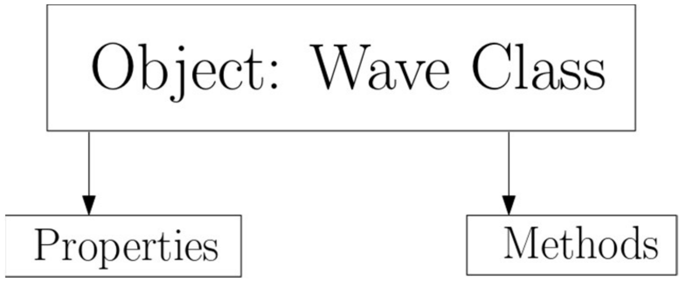
Online Training Materials

PRESENTED BY

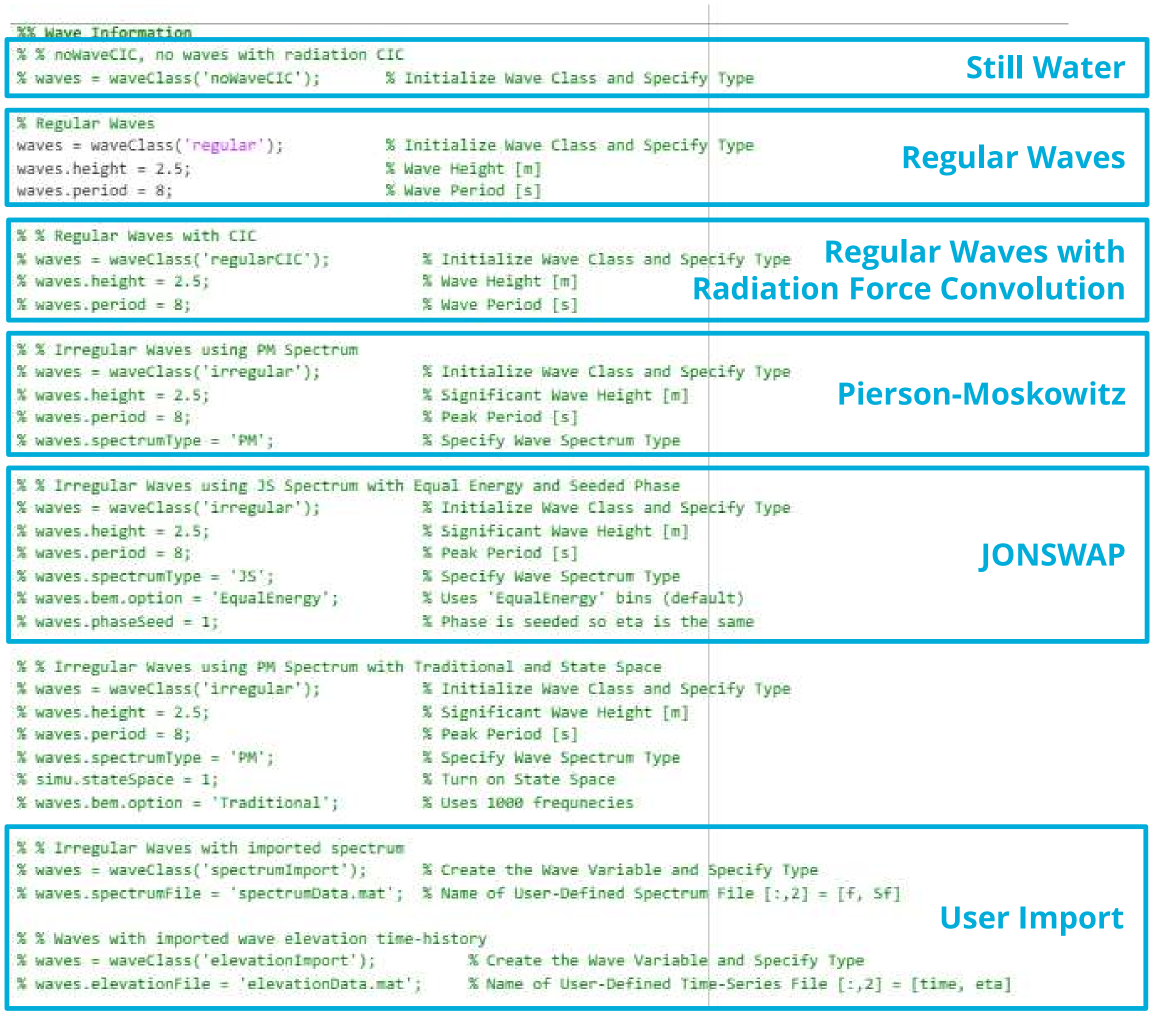
Jeff Grasberger, Sandia

Wave Class

Wave Class: Overview



Choice of wave inputs to the system



```
%% Wave Information
% % noWaveCIC, no waves with radiation CIC
% waves = waveClass('noWaveCIC');       % Initialize Wave Class and Specify Type
```

Still Water

```
% Regular Waves
waves = waveClass('regular');           % Initialize Wave Class and Specify Type
waves.height = 2.5;                     % Wave Height [m]
waves.period = 8;                       % Wave Period [s]
```

Regular Waves

```
% % Regular Waves with CIC
% waves = waveClass('regularCIC');          % Initialize Wave Class and Specify Type
% waves.height = 2.5;                       % Wave Height [m]
% waves.period = 8;                         % Wave Period [s]
```

Regular Waves with Radiation Force Convolution

```
% % Irregular Waves using PM Spectrum
% waves = waveClass('irregular');           % Initialize Wave Class and Specify Type
% waves.height = 2.5;                       % Significant Wave Height [m]
% waves.period = 8;                         % Peak Period [s]
% waves.spectrumType = 'PM';                % Specify Wave Spectrum Type
```

Pierson-Moskowitz

```
% % Irregular Waves using JS Spectrum with Equal Energy and Seeded Phase
% waves = waveClass('irregular');           % Initialize Wave Class and Specify Type
% waves.height = 2.5;                       % Significant Wave Height [m]
% waves.period = 8;                         % Peak Period [s]
% waves.spectrumType = 'JS';                % Specify Wave Spectrum Type
% waves.bem.option = 'EqualEnergy';         % Uses 'EqualEnergy' bins (default)
% waves.phaseSeed = 1;                      % Phase is seeded so eta is the same
```

JONSWAP

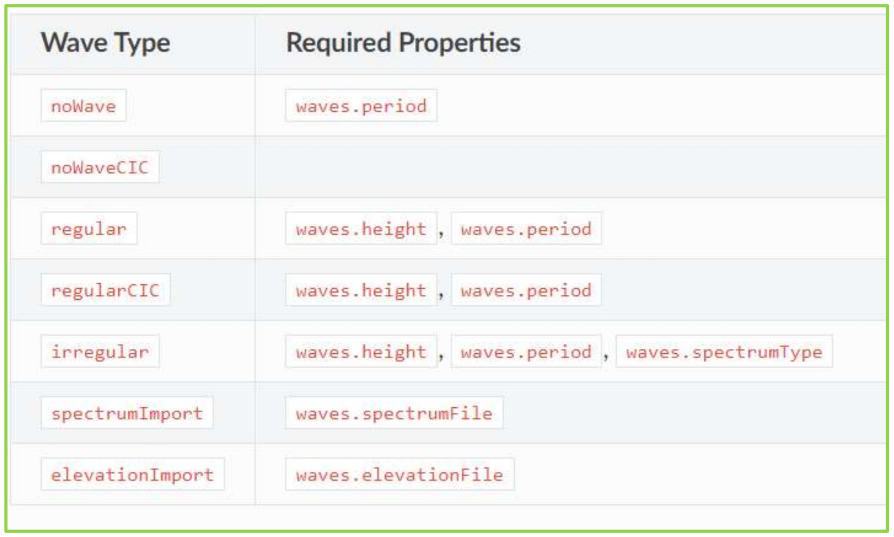
```
% % Irregular Waves using PM Spectrum with Traditional and State Space
% waves = waveClass('irregular');           % Initialize Wave Class and Specify Type
% waves.height = 2.5;                       % Significant Wave Height [m]
% waves.period = 8;                         % Peak Period [s]
% waves.spectrumType = 'PM';                % Specify Wave Spectrum Type
% simu.stateSpace = 1;                      % Turn on State Space
% waves.bem.option = 'Traditional';         % Uses 1000 frequnecies
```

```
% % Irregular Waves with imported spectrum
% waves = waveClass('spectrumImport');      % Create the Wave Variable and Specify Type
% waves.spectrumFile = 'spectrumData.mat';  % Name of User-Defined Spectrum File [:,2] = [f, Sf]

% % Waves with imported wave elevation time-history
% waves = waveClass('elevationImport');          % Create the Wave Variable and Specify Type
% waves.elevationFile = 'elevationData.mat';     % Name of User-Defined Time-Series File [:,2] = [time, eta]
```

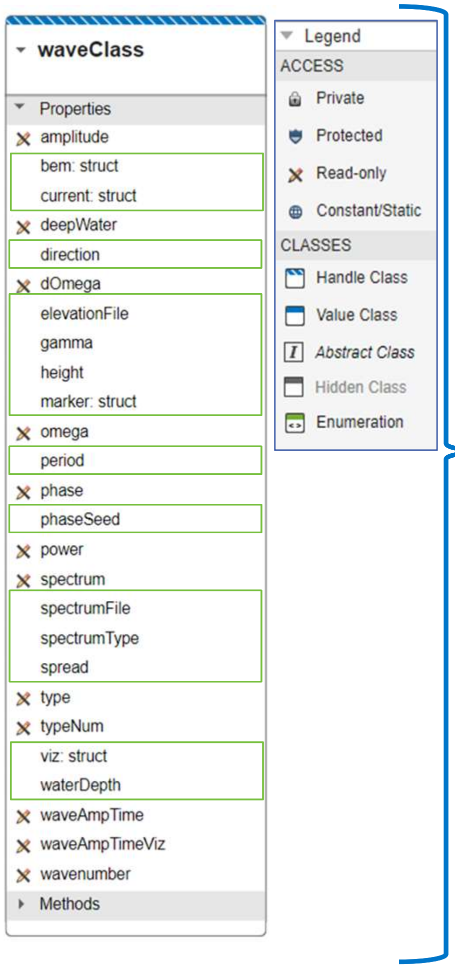
User Import

Wave Class: Properties

Object: Wave Class
Properties
Methods



Wave Type	Required Properties
`noWave`	`waves.period`
`noWaveCIC`	
`regular`	`waves.height` , `waves.period`
`regularCIC`	`waves.height` , `waves.period`
`irregular`	`waves.height` , `waves.period` , `waves.spectrumType`
`spectrumImport`	`waves.spectrumFile`
`elevationImport`	`waves.elevationFile`



waveClass

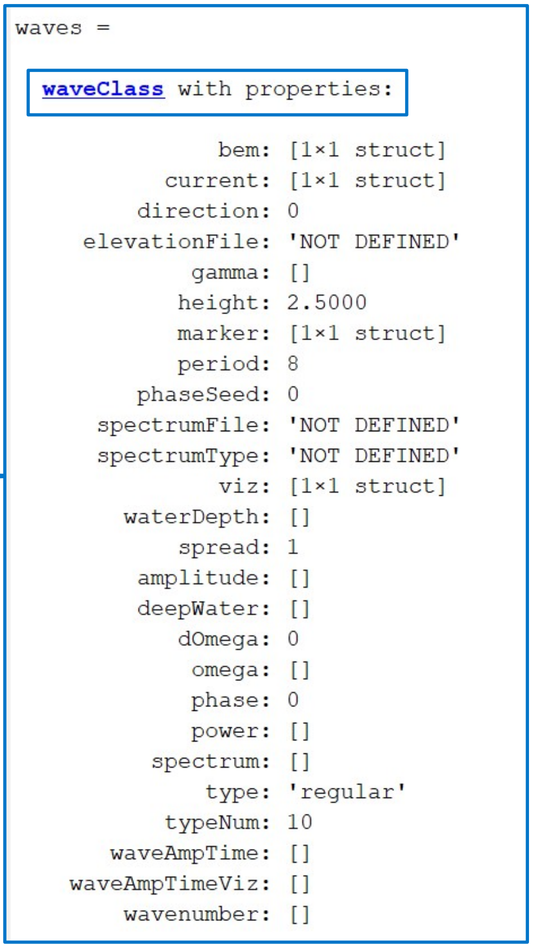
Properties
- ✕ amplitude
- bem: struct
- current: struct
- ✕ deepWater
- direction
- ✕ dOmega
- elevationFile
- gamma
- height
- marker: struct
- ✕ omega
- period
- ✕ phase
- phaseSeed
- ✕ power
- ✕ spectrum
- spectrumFile
- spectrumType
- spread
- ✕ type
- ✕ typeNum
- viz: struct
- waterDepth
- ✕ waveAmpTime
- ✕ waveAmpTimeViz
- ✕ wavenumber

Methods

Legend

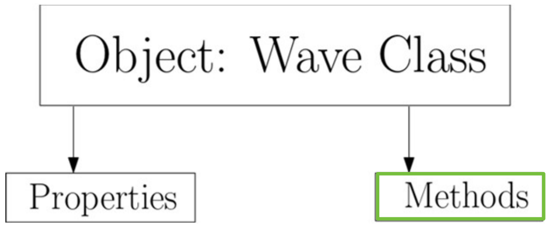
ACCESS	
🔒	Private
🛡	Protected
✕	Read-only
●	Constant/Static

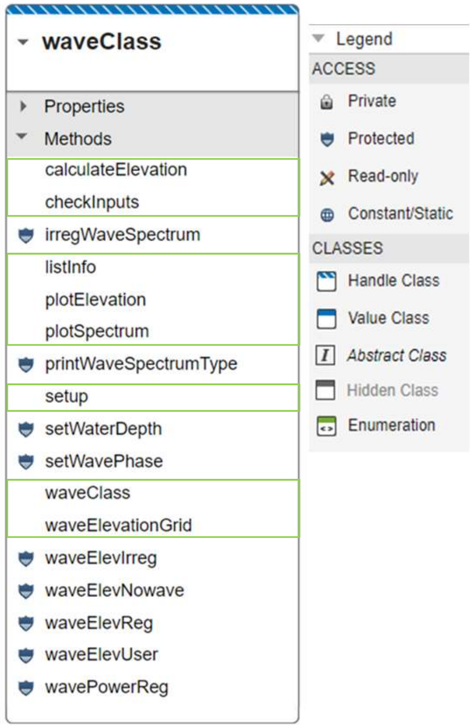
CLASSES	
▣	Handle Class
▭	Value Class
I	*Abstract Class*
▭	Hidden Class
▤	Enumeration



```
waves =

  waveClass with properties:

              bem: [1×1 struct]
          current: [1×1 struct]
        direction: 0
    elevationFile: 'NOT DEFINED'
            gamma: []
           height: 2.5000
           marker: [1×1 struct]
           period: 8
        phaseSeed: 0
     spectrumFile: 'NOT DEFINED'
     spectrumType: 'NOT DEFINED'
              viz: [1×1 struct]
       waterDepth: []
           spread: 1
        amplitude: []
        deepWater: []
           dOmega: 0
            omega: []
            phase: 0
            power: []
         spectrum: []
             type: 'regular'
          typeNum: 10
      waveAmpTime: []
   waveAmpTimeViz: []
       wavenumber: []
```

Wave Class: Methods





waveClass

- Properties
- Methods
  - calculateElevation
  - checkInputs
  - irregWaveSpectrum
  - listInfo
  - plotElevation
  - plotSpectrum
  - printWaveSpectrumType
  - setup
  - setWaterDepth
  - setWavePhase
  - waveClass
  - waveElevationGrid
  - waveElevIrreg
  - waveElevNowave
  - waveElevReg
  - waveElevUser
  - wavePowerReg

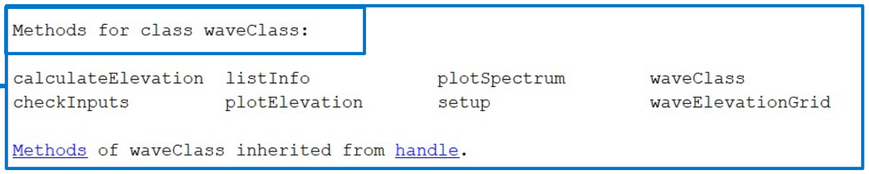
Legend

ACCESS

- Private
- Protected
- Read-only
- Constant/Static

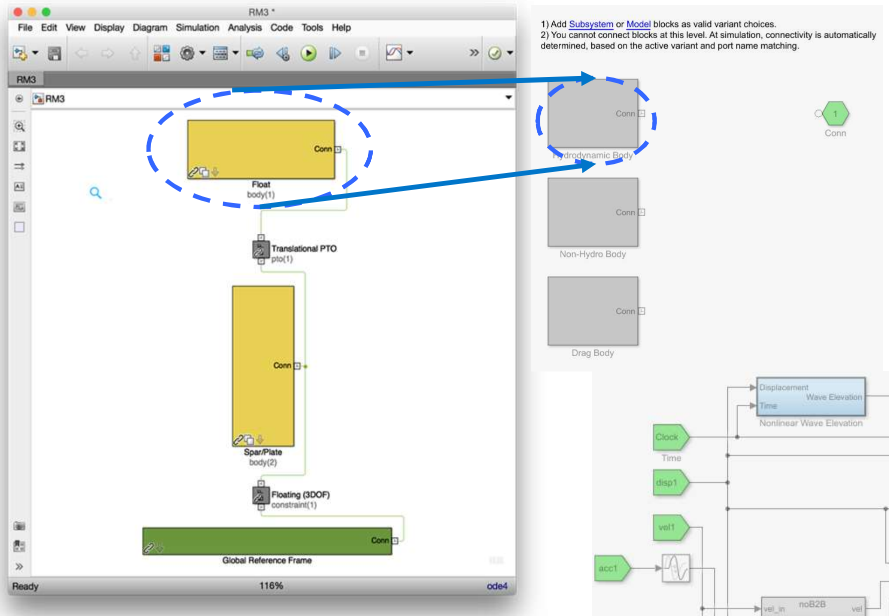
CLASSES

- Handle Class
- Value Class
- *Abstract Class*
- Hidden Class
- Enumeration

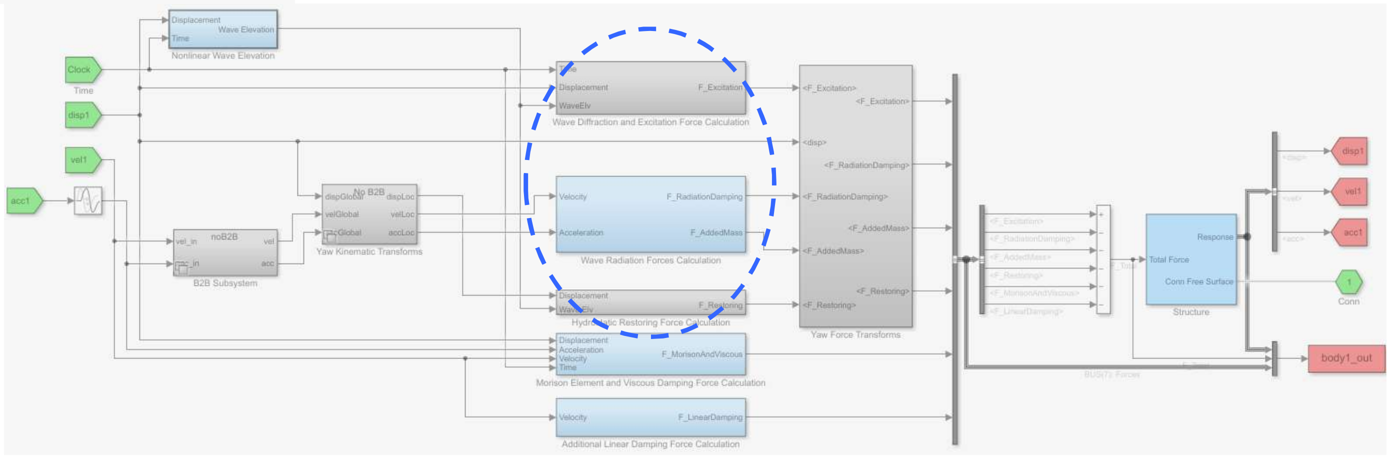


```
Methods for class waveClass:

calculateElevation  listInfo          plotSpectrum      waveClass
checkInputs         plotElevation     setup             waveElevationGrid

Methods of waveClass inherited from handle.
```

Wave Class Simulink



1) Add Subsystem or Model blocks as valid variant choices.
2) You cannot connect blocks at this level. At simulation, connectivity is automatically determined, based on the active variant and port name matching.


Nonlinear Wave Elevation
Clock
Time
disp1
vel1
acc1
Wave Diffraction and Excitation Force Calculation
Wave Radiation Forces Calculation
Hydrostatic Restoring Force Calculation
Morison Element and Viscous Damping Force Calculation
Additional Linear Damping Force Calculation
B2B Subsystem
Yaw Kinematic Transforms
Yaw Force Transforms
Structure
body1_out

Simulink Applies (based on waveClass inputs):

- Wave Diffraction and Excitation Force
- Wave Radiation Force
  - Constant Coefficient
  - Convolution Integral
  - State Space
- Hydrostatic Restoring Force

Thank you

For more information please visit the WEC-Sim website:

http://wec-sim.github.io/WEC-Sim

If you have questions on this presentation please reach out to any of the WEC-Sim Developers on GitHub:

https://github.com/WEC-Sim/WEC-Sim

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308.

Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Water Power Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.