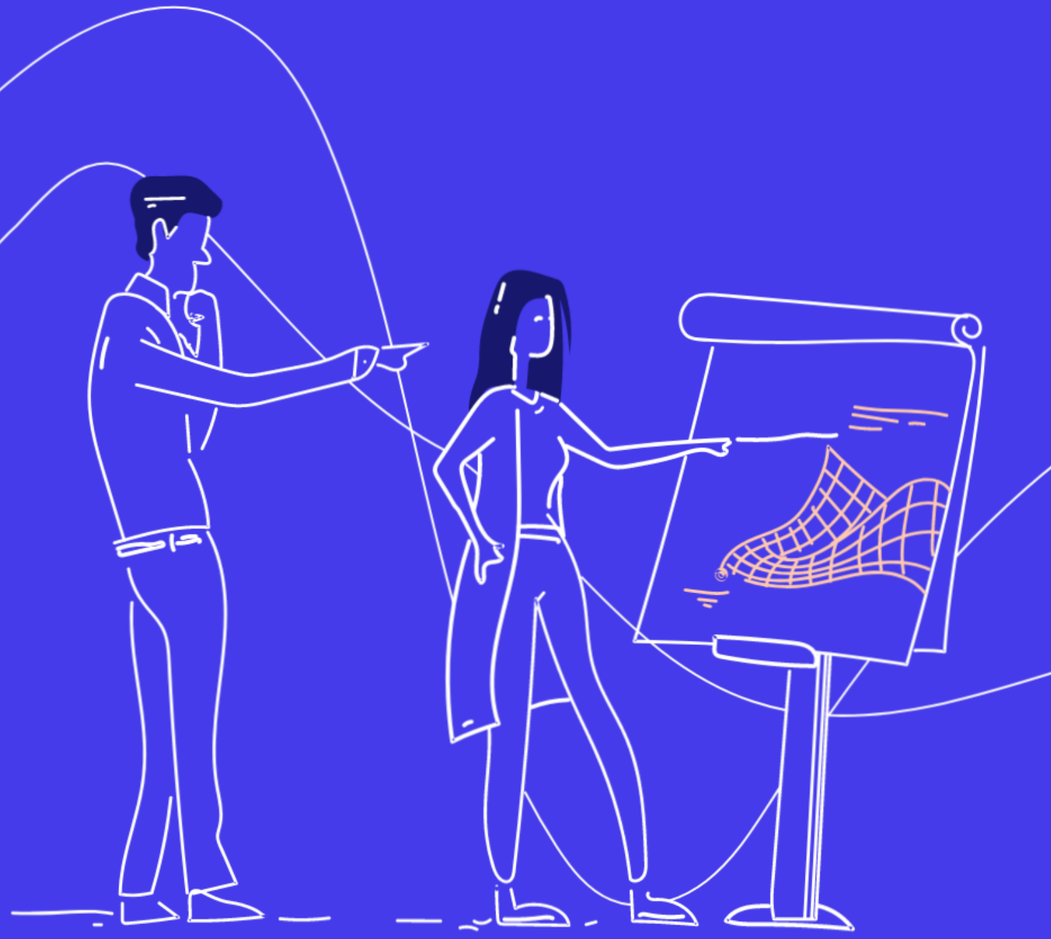
sciengineer

MATLAB to C with MATLAB Coder



SciEngineer's training courses are designed to help organizations and individuals close skills gaps, keep up-to-date with the industry-accepted best practices and achieve the greatest value from MathWorks® and COMSOL® Products.

MATLAB to C with MATLAB Coder

This two-day course covers C code generation from MATLAB code using MATLAB Coder. The focus is on making existing MATLAB code compliant, generating C code that meets optimization requirements, and integrating generated code with external modules. Topics include: preparing MATLAB code for code generation, working with fixed-size and variable-size data, integrating with external code and optimizing generated code.

Prerequisites

MATLAB Fundamentals and knowledge of C programming language

TOPICS

Day 1

- Code Generation with MATLAB Coder
- Preparing MATLAB Code for Code Generation
- Working with Fixed-Size Data
- Working with Variable-Size Data

Day 2

- Working with Global Data, Structures, and Cell Arrays
- Integrating with External Code
- Optimizing Generated Code

Code Generation with MATLAB Coder

OBJECTIVE: Become familiar with MATLAB Coder and its applications.

- MATLAB Coder overview
- Workflow for generating C code from MATLAB code
- Generating C code
- Verifying generated code
- Navigating generated code

Preparing MATLAB Code for Code Generation

OBJECTIVE: Use MATLAB Coder coding standards to write MATLAB code that is ready for code generation.

- Translating MATLAB code into C code
- Calling unsupported MATLAB functions
- Preparing existing MATLAB code
- Code preparation workflows

Working with Fixed-Size Data

OBJECTIVE: Generate C code from MATLAB code that has fixed-size or constant inputs.

- Data characteristics overview
- Specifying fixed-size, top-level inputs
- Specifying constant top-level inputs

Working with Variable-Size Data

OBJECTIVE: Generate C code from MATLAB code that has variable-size inputs or local data.

- Specifying variable-size, top-level inputs
- Specifying variable-size local data
- Reusing variables

Optimizing Generated Code

OBJECTIVE: Use various options and techniques to optimize generated code.

- Code optimization with loop unrolling and null initialization
- Function inlining and file partitioning
- Configuration objects
- Removing unnecessary code
- Naming conventions in generated code
- Converting a project to a script

Integrating with External Code

OBJECTIVE: Integrate generated C code from MATLAB Coder with external C code.

- Code integration overview
- Entry points to generated code
- Integrating external C code using MATLAB Coder interface
- Integrating external C code using an external IDE
- Calling external C functions
- Code verification and profiling
- Source code debugging

Working with Global Data, Structures, and Cell Arrays

OBJECTIVE: Generate C code from MATLAB code that contains persistent data, global variables, input structures, or cell arrays.

- Persistent variables
- Global variables
- Working with structures
- Cell arrays in generated code
- Passing arguments by reference

sciengineer

Expand your knowledge