Category Theory is an abstract branch of mathematics that provides a language for dealing with many different types of mathematical structure. As a way of reasoning about structure and identity it is remarkably elegant. It also offers excellent opportunities for automating operations that manipulate structures. For this reason, it has been widely explored as a way of representing structure in software specifications, and for expressing the relationships between specifications and their refinements. A clear advantage of using category theory is that it separates out the representation of structure from the specification language being used, thus keeping the specifications languages simpler.

Unfortunately, the elegance of category theory comes with a price. The very abstractness of category theory can make it hard to understand, so that arguments and proofs that use the language of category theory can be hard to follow.

In this mini-tutorial, I will attempt to de-mystify category theory for those that have always wondered what it is all about but were afraid to ask. Using concrete examples, I will explain the basics of category theory, and illustrate the main constructs that are relevant to its application in software specification. I will then proceed to demonstrate how these constructs can be applied, concentrating primarily on the use of specification morphisms and what you can do with them. I will end with a survey of applications of category theory to automated software engineering, along with some comments on the benefits that have been derived. This mini-tutorial will assume no prior knowledge of category theory, mathematics, greek symbols, etc. Bring a sharp pencil and plenty of paper.