

HOTT EXERCISES IN AGDA

ERIC BAILEY

CONTENTS

1. Chapter 1	2
1.1. Exercise 1	2

1. CHAPTER 1

1.1. Exercise 1.

Use the same options as HoTT-Agda.

```
{-# OPTIONS --without-K --rewriting #-}
```

Declare the module.

```
module HoTT.Chapter1.Ex1 where
```

Import the propositional equality module for \equiv .

```
open import Relation.Binary.PropositionalEquality
```

Given functions $f : A \rightarrow B$ and $g : B \rightarrow C$, define their **composite** $g \circ f : A \rightarrow C$.

```
_o_ : ∀ {A B C : Set} → (B → C) → (A → B) → A → C
```

```
g o f = x → g (f x)
```

Prove that function composition is associative, i.e.

$$h \circ (g \circ f) \equiv (h \circ g) \circ f$$

```
o-assoc : ∀ {A B C D : Set} {f : A → B} {g : B → C} {h : C → D} →
  h o (g o f) ≡ (h o g) o f
```

```
o-assoc = refl
```