

# Slices in Go

---

Array	Slice
<ul style="list-style-type: none"><li>● Has a <b>fixed length</b> defined at compile time;</li><li>● The length of an array is part of its type, defined at compile time and cannot be changed;</li><li>● By default an uninitialized array has all elements equal to zero;</li></ul>	<ul style="list-style-type: none"><li>● Has a <b>dynamic length</b> (it can shrink or grow);</li><li>● The length of a slice is <b>not part</b> of its type and it belongs to runtime;</li><li>● An uninitialized slice is equal to <b>nil</b> (its zero value is nil).</li></ul>

- Both a slice and an array can contain only the same type of elements;
- We can create a keyed slice like a keyed array;

# Slice's Backing (Underlying) Array

---

- When creating a slice, behind the scenes Go creates a **hidden** array called **Backing Array**.
- The backing array stores the elements, not the slice.
- Go implements a slice as a data structure called **slice header**.

**Slice Header** contains 3 fields:

1. **the address** of the backing array (pointer).
  2. **the length** of the slice. **len()** returns it.
  3. **the capacity** of the slice. **cap()** returns it.
- 
- Slice Header is the runtime representation of a slice.
  - A **nil slice** doesn't have backing array.