Quick Check vs. Scala Check

```
let test_neutral_right =
  QCheck.Test.make
    ~name:"forall li. append li [] = li"
    QCheck.(list int)
    (fun li -> List.append li [] = li)
```

```
property("forall li. append li [] = li") =
    forAll {
      la: List[Int] => append(la, Nil) == la
    }
```

Quick Check vs. Scala Check

```
let test_head =
    let open QCheck in
    Test.make
      ~name:"forall la lb. head (append la lb) = head la"
      (pair (list int) (list int))
      (fun (la, lb) ->
        assume (la <> []);
        (List.hd (List.append la lb) = List.hd la))
```

```
property("forall la lb. head (append la lb) = head la") =
    forAll {
      (la: List[Int], lb: List[Int]) =>
              la.nonEmpty ==>
              (append(la,lb).head == la.head)
    }
```

Quick Check vs. Scala Check

```
let test_sum_even =
  QCheck.Test.make
    ~name:"even lists have an even sum"
    QCheck.(list (map (fun n -> 2*n) int))
    (fun li ->
       QCheck.assume (List.for_all even li);
       even (sum li))
```

```
property("even lists have an even sum") =
    forAll(Gen.posNum[Int].map(n => n*2)) {
      li: List[Int] =>
              li.forall(even) ==>
              even(sum(li))
    }
```