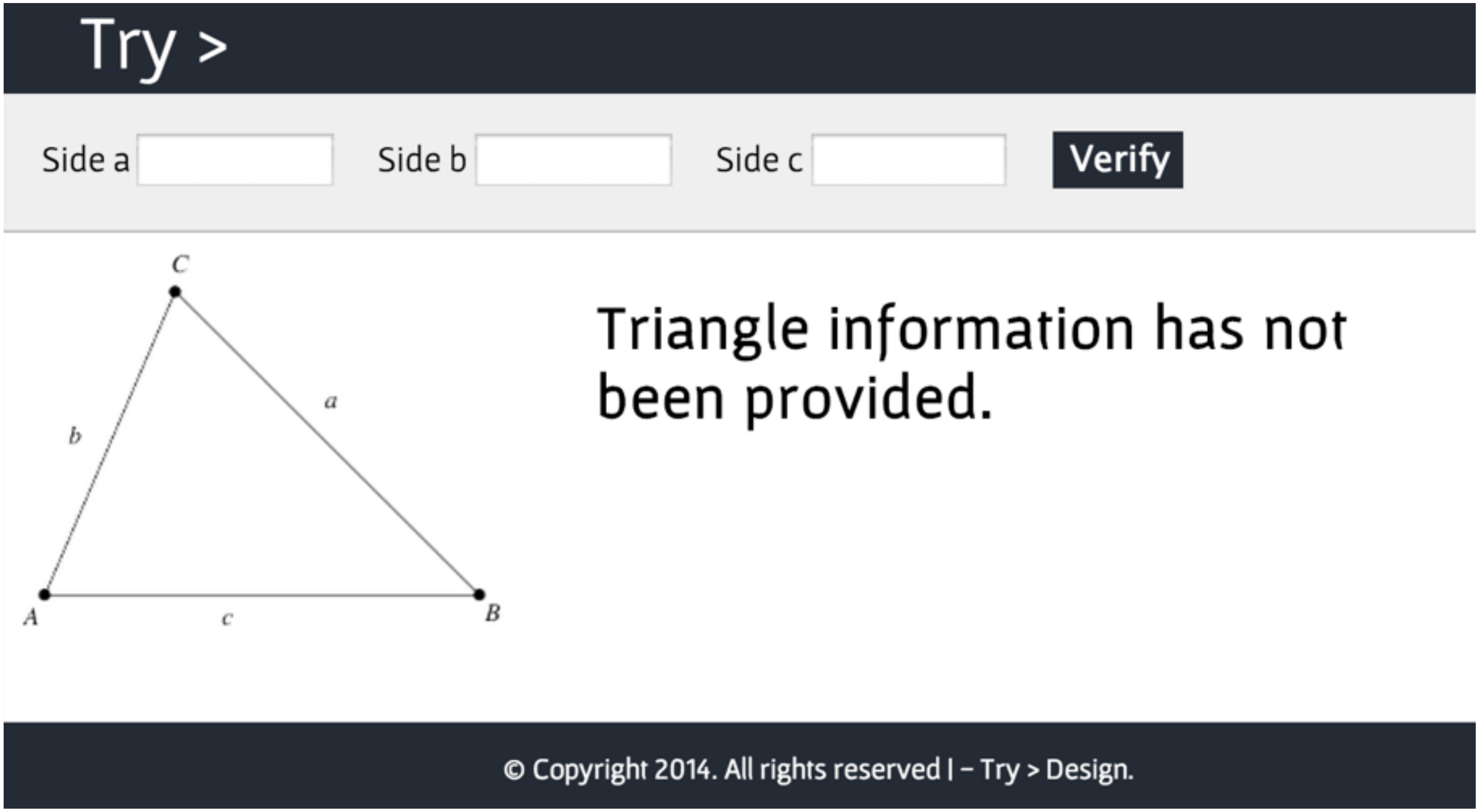
expect(“jasmine”)

behaviour-driven javascript testing

We can ask users to test



Try >

Side a

Side b

Side c

Verify

Triangle information has not been provided.

© Copyright 2014. All rights reserved | – Try > Design.

Test cases

Test	Input	Expected output
Successful input error handling	The user enters a non valid number	error message
Successfully identifying valid triangles	Valid triangle sides (e.g., 2, 3, 4)	message confirming that the triangle is valid
Successfully identifying non-valid triangles	Valid triangle sides (e.g., 1, 1, 10)	message explaining that the input is not a valid triangle
Successfully determining a equilateral triangle	Valid equilateral triangle (e.g., 3, 3, 3)	message confirming that the triangle is *equilateral*
Successfully determining an isosceles triangle	Valid isosceles triangle (e.g., 4, 4, 1)	message confirming that the triangle is *isosceles*
Successfully determining a scalene triangle	Valid scalene triangle (e.g., 3, 4, 5)	message confirming that the triangle is *scalene*

The TryAngle API

```
/* Check whether the input sides belong to a triangle.
 * return true if the sides a, b, c correspond to a valid triangle,
 *           false otherwise
 */
TryAngle.isTriangle(a, b, c);

/*  Get the type of triangle by side length:
 *  return TryAngle.SIDE_EQUILATERAL
 *             TryAngle.SIDE_ISOSCELES
 *             TryAngle.SIDE_SCALENE
 */
TryAngle.getTypeBySidesLength(a, b, c);

/*  Get the type of triangle by angle:
 *  return TryAngle.ANGLE_ACUTE
 *             TryAngle.ANGLE_OBTUSE
 *             TryAngle.ANGLE_RIGHT
 */
TryAngle.getTypeByAngles(a, b, c);
```

Jasmine

```
describe("TryAngle", function(){

    it ("should allow…", function(){
        expect(property).toBe(value)
    });

});
```

Jasmine

```
describe("TryAngle", function(){

    it ("should allow...", function(){
        expect(property).toBe(value)
    });

});
```

expectation / assertion

actual value

expected value

matcher

===

A spec with all true expectations is a passing spec. A spec with one or more false expectations is a failing spec.

Some useful matchers

expect(input).**toBe**(value)

expect(input).**not**.**toBe**(value)

expect(input).**toEqual**(value)