The Big Picture: What Is Programming?

INTRODUCTION

Simon Allardice
STAFF AUTHOR, PLURALSIGHT
@allardice www.pluralsight.com

Source Code

COBOL

```
PROCEDURE DIVISION USING L-Input-Date-DT
                   RETURNING L-Output-Day-NUM.
000-Main SECTION.

    EVALUATE RETURN-CODE
    WHEN 7
        IF TEST-DAY-YYYYDDD(L-Input-Date-DT) > 0
            MOVE 0 TO L-Output-Day-NUM
            GOBACK
        END-IF
        MOVE DATE-OF-INTEGER(INTEGER-OF-DAY(L-Input-Date-DT))
             TO WS-Input-Date-DT
    WHEN 8
        IF TEST-DATE-YYYYMMDD(L-Input-Date-DT) > 0
            MOVE 0 TO L-Output-Day-NUM
            GOBACK
        END-IF
```

Python

```
def sing(b, end):
    print(b or 'no more','bottle'+('s' if b-1 else ''), end)

for i in range(99, 0, -1):
    sing(i, 'of beer on the wall,')
    sing(i, 'of beer,')
    print('take one down, pass it around,')
    sing(i-1, 'of beer on the wall.\n')
```

PHP

```
function last_friday_of_month($year, $month) {

  $day = 0;

  while(True) {

    $last_day = mktime(0, 0, 0, $month+1, $day, $year);

    if (date("w", $last_day) == 5) {
      return date("Y-m-d", $last_day);
    }

    $day -= 1;
  }

}
```

PHP

```
function last_friday_of_month($year, $month) {

  $day = 0;

  while(True) {

    $last_day = mktime(0, 0, 0, $month+1, $day, $year);

    if (date("w", $last_day) == 5) {
      return date("Y-m-d", $last_day);
    }

    $day -= 1;
  }

}
```

PHP

```
function last_friday_of_month($year, $month) {

  $day = 0;

  while(True) {

    $last_day = mktime(0, 0, 0, $month+1, $day, $year);

    if (date("w", $last_day) == 5) {
      return date("Y-m-d", $last_day);
    }

    $day -= 1;
  }

}
```

Visual Basic

```
For i = 0 To 11
    Console.Write "On the " & days(i) & " day of Christmas"
    Console.Write "My true love sent to me:"
    If i = 0 Then
        Console.Write gifts(i)
    Else
        For j = i To 0 Step - 1
            If j = 0 Then
                Console.Write "and " & gifts(0)
            Else
                Console.Write gifts(j)
                Console.WriteLine
            End If
        Next
    End If
        Console.WriteLine
Next
```

JavaScript Source Code

```javascript
var last_friday_of_month, print_last_fridays_of_month;

last_friday_of_month = function(year, month) {
  var i, last_day;
  i = 0;
  while (true) {
    last_day = new Date(year, month, i);
    if (last_day.getDay() === 5) {
      return last_day.toDateString();
    }
    i -= 1;
  }
};

print_last_fridays_of_month = function(year) {
  var month, results;
  results = [];
  for (month = 1; month <= 12; ++month) {
    results.push(console.log(last_friday_of_month(year, month)));
  }
  return results;
};

(function() {
  var year;
  year = parseInt(process.argv[2]);
  return print_last_fridays_of_month(year);
})();
```

Swift Source Code

```swift
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}

class Character {

    // MARK: Dealing with fire

    private var isBurning = false
    private var isInvincible = false

    private var fireEmitter: ParticleEmitter! = nil
    private var smokeEmitter: ParticleEmitter! = nil
    private var whiteSmokeEmitter: ParticleEmitter! = nil

    func haltFire() {
        if isBurning {
            isBurning = false

            node.runAction(SCNAction.sequence([
                SCNAction.playAudioSource(haltFireSound, waitForCompletion: true),
                SCNAction.playAudioSource(reliefSound, waitForCompletion: false)])
            )
```

Python Source Code

```python
    def enabled(self):
        return self._ydl.params.get('cachedir') is not False

    def store(self, section, key, data, dtype='json'):
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:
            try:
                os.makedirs(os.path.dirname(fn))
            except OSError as ose:
                if ose.errno != errno.EEXIST:
                    raise
            write_json_file(data, fn)
        except Exception:
            tb = traceback.format_exc()
            self._ydl.report_warning(
```

C# Source Code

```csharp
internal static class Program
    {
        private static IEnumerable<DateTime> LastFridaysOfYear(int year)
        {
            for (var month = 1; month <= 12; month++)
            {
                var date = new DateTime(year, month,1).AddMonths(1).AddDays(-1);
                while (date.DayOfWeek != DayOfWeek.Friday)
                {
                    date = date.AddDays(-1);
                }
                yield return date;
            }
        }

        private static void Main(string[] arguments)
        {
            int year;
            var argument = arguments.FirstOrDefault();
```

JavaScript Source Code

```javascript
var last_friday_of_month, print_last_fridays_of_month;

last_friday_of_month = function(year, month) {
  var i, last_day;
  i = 0;
  while (true) {
    last_day = new Date(year, month, i);
    if (last_day.getDay() === 5) {
      return last_day.toDateString();
    }
    i -= 1;
  }
};

print_last_fridays_of_month = function(year) {
  var month, results;
  results = [];
  for (month = 1; month <= 12; ++month) {
    results.push(console.log(last_friday_of_month(year, month)));
  }
```

JavaScript

```javascript
  return print_last_fridays_of_month(year);
```

Swift Source Code

```swift
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}

class Character {

    // MARK: Dealing with fire

    private var isBurning = false
    private var isInvincible = false

    private var fireEmitter: ParticleEmitter! = nil
    private var smokeEmitter: ParticleEmitter! = nil
    private var whiteSmokeEmitter: ParticleEmitter! = nil

    func haltFire() {
```

Swift

```swift
            )
```

Python Source Code

```python
    def enabled(self):
        return self._ydl.params.get('cachedir') is not False

    def store(self, section, key, data, dtype='json'):
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:
```

Python

```python
        except Exception:
            tb = traceback.format_exc()
```

C# Source Code

```csharp
internal static class Program
    {
        private static IEnumerable<DateTime> LastFridaysOfYear(int year)
        {
            for (var month = 1; month <= 12; month++)
            {
                var date = new DateTime(year, month,1).AddMonths(1).AddDays(-1);
                while (date.DayOfWeek != DayOfWeek.Friday)
                {
                    date = date.AddDays(-1);
                }
```

C#

```csharp
        {
            int year;
            var argument = arguments.FirstOrDefault();
```

JavaScript Source Code

```javascript
var last_friday_of_month, print_last_fridays_of_month;

last_friday_of_month = function(year, month) {
  var i, last_day;
  i = 0;
  while (true) {
    last_day = new Date(year, month, i);
    if (last_day.getDay() === 5) {
      return last_day.toDateString();
    }
    i -= 1;
  }
};

print_last_fridays_of_month = function(year) {
  var month, results;
  results = [];
  for (month = 1; month <= 12; ++month) {
    results.push(console.log(last_friday_of_month(year, month)));
  }
```

JavaScript Source Code

```javascript
  return print_last_fridays_of_month(year);
```

Swift Source Code

```swift
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}

class Character {

    // MARK: Dealing with fire

    private var isBurning = false
    private var isInvincible = false

    private var fireEmitter: ParticleEmitter! = nil
    private var smokeEmitter: ParticleEmitter! = nil
    private var whiteSmokeEmitter: ParticleEmitter! = nil

    func haltFire() {
```

Swift Source Code

```swift
            )
```

Python Source Code

```python
    def enabled(self):
        return self._ydl.params.get('cachedir') is not False

    def store(self, section, key, data, dtype='json'):
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:
```

Python Source Code

```python
        except Exception:
            tb = traceback.format_exc()
```

C# Source Code

```csharp
internal static class Program
    {
        private static IEnumerable<DateTime> LastFridaysOfYear(int year)
        {
            for (var month = 1; month <= 12; month++)
            {
                var date = new DateTime(year, month,1).AddMonths(1).AddDays(-1);
                while (date.DayOfWeek != DayOfWeek.Friday)
                {
                    date = date.AddDays(-1);
                }
```

C# Source Code

```csharp
        {
            int year;
            var argument = arguments.FirstOrDefault();
```

JavaScript Source Code

```javascript
var last_friday_of_month, print_last_fridays_of_month;

last_friday_of_month = function(year, month) {
  var i, last_day;
  i = 0;
  while (true) {
    last_day = new Date(year, month, i);
    if (last_day.getDay() === 5) {
      return last_day.toDateString();
    }
    i -= 1;
  }
};

print_last_fridays_of_month = function(year) {
  var month, results;
  results = [];
  for (month = 1; month <= 12; ++month) {
    results.push(console.log(last_friday_of_month(
  }
```

```javascript
  return print_last_fridays_of_month(year);
```

JavaScript Source Cod

Swift Source Code

```swift
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}
```

```swift
ire

 false
 = false

 ParticleEmitter! = nil
: ParticleEmitter! = nil
itter: ParticleEmitter! = nil
```

rce Code

Dictionary

Search

source code |ˈsôrs ˌkōd|

noun *computing*

A collection of computer instructions written using some human-readable computer programming language, usually as plain text.

Python Source Code

```python
    def enabled(self):
        return self._ydl.params.get('cachedir') is

    def store(self, section, key, data, dtype='jso
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:
```

```python
        except Exception:
            tb = traceback.format_exc()
```

Python Source Code

C# Source Code

```csharp
ram

umerable<DateTime> LastFridaysOfYear(int year)

 = 1; month <= 12; month++)

 new DateTime(year, month,1).AddMonths(1).AddDays(-1);
e.DayOfWeek != DayOfWeek.Friday)
{
    date = date.AddDays(-1);
}
```

```csharp
{
    int year;
    var argument = arguments.FirstOrDefault();
```

C# Source Code

Recipe X

Recipe X

what ingredients?

Recipe X

what ingredients?

what quantities?

Recipe X

what ingredients?

what quantities?

step 1...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

step 4...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

step 4...

step 3...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

step 4...

step 3...

step 5...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

step 4...

step 3...

step 5...

step 6...

Recipe X

what ingredients?

what quantities?

step 1...

step 2...

step 3...

step 4...

step 5...

step 6...

Programmer's Text Editors

Programmer's Text Editors

Notepad++
Sublime Text
Brackets
Atom
jEdit
Vim
Emacs
TextWrangler
UltraEdit
BBEdit
Coda
EditPad
BlueFish
KomodoEdit
GNU nano

Most Programmers use a **Monospaced Font**

Most Programmers use a **Monospaced Font**

Monospaced fonts

(i.e. Menlo, Consolas, Courier)

Every character is the same width

Most Programmers use a **Monospaced Font**

Monospaced fonts

(i.e. Menlo, Consolas, Courier)

Every character is the same width

Proportional fonts

(i.e. Arial, Verdana, Times)

Different characters have different widths

Most Programmers use a **Monospaced Font**

Monospaced fonts

(i.e. Menlo, Consolas, Courier)

Every character is the same width

Proportional fonts

(i.e. Arial, Verdana, Times)

Different characters have different widths

Most Programmers use a **Monospaced Font**

Monospaced fonts

(i.e. Menlo, Consolas, Courier)

Every character is the same width

Proportional fonts

(i.e. Arial, Verdana, Times)

Different characters have different widths

An exact layout makes code easier to read

```
void key_set_timeout(struct key *key, unsigned timeout)
{
  struct timespec now;
  time_t expiry = 0;

  /* make the changes with the locks held to prevent races */
  down_write(&key->sem);

  if (timeout > 0) {
    now = current_kernel_time();
    expiry = now.tv_sec + timeout;
  }

  key->expiry = expiry;
  key_schedule_gc(key->expiry + key_gc_delay);

  up_write(&key->sem);
}
```

Getting the Computer to Understand You

C# Source Code

```
internal static class Program
    {
        private static IEnumerable<DateTime> LastFridaysOfYear(int year)
        {
            for (var month = 1; month <= 12; month++)
            {
                var date = new DateTime(year, month,1).AddMonths(1).AddDays(-1);
                while (date.DayOfWeek != DayOfWeek.Friday)
                {
                    date = date.AddDays(-1);
                }
                yield return date;
            }
        }

        private static void Main(string[] arguments)
        {
            int year;
            var argument = arguments.FirstOrDefault();
            if (string.IsNullOrEmpty(argument) ||
               !int.TryParse(argument, out year))
            {
                year = DateTime.Today.Year;
            }

            foreach (var date in LastFridaysOfYear(year))
            {
```

Swift Source Code

```
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}

class Character {

    // MARK: Dealing with fire

    private var isBurning = false
    private var isInvincible = false

    private var fireEmitter: ParticleEmitter! = nil
    private var smokeEmitter: ParticleEmitter! = nil
    private var whiteSmokeEmitter: ParticleEmitter! = nil

    func haltFire() {
        if isBurning {
            isBurning = false

            node.runAction(SCNAction.sequence([
                SCNAction.playAudioSource(haltFireSound, waitForCompletion: true),
                SCNAction.playAudioSource(reliefSound, waitForCompletion: false)])
            )
```

Python Source Code

```
    def enabled(self):
        return self._ydl.params.get('cachedir') is not False

    def store(self, section, key, data, dtype='json'):
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:
            try:
                os.makedirs(os.path.dirname(fn))
            except OSError as ose:
                if ose.errno != errno.EEXIST:
                    raise
            write_json_file(data, fn)
        except Exception:
            tb = traceback.format_exc()
            self._ydl.report_warning(
```

ruby example

```
  class FrontmatterDefaults
    # Initializes a new instance.
    def initialize(site)
      @site = site
    end

    def update_deprecated_types(set)
      return set unless set.key?('scope') && set['scope'].key?('type')

      set['scope']['type'] =
        case set['scope']['type']
        when 'page'
          Deprecator.defaults_deprecate_type('page', 'pages')
          'pages'
        when 'post'
          Deprecator.defaults_deprecate_type('post', 'posts')
          'posts'
        when 'draft'
          Deprecator.defaults_deprecate_type('draft', 'drafts')
          'drafts'
        else
```

C# Source Code

```
internal static class Program
    {
        private static IEnumerable<DateTime> LastFridaysOfYear(int year)
        {
            for (var month = 1; month <= 12; month++)
            {
                var date = new DateTime(year, month,1).AddMonths(1).AddDays(-1);
                while (date.DayOfWeek != DayOfWeek.Friday)
                {
                    date = date.AddDays(-1);
                }
                yield return date;
            }
        }

        private static void Main(string[] arguments)
        {
            int year;
            var argument = arguments.FirstOrDefault();

            foreach (var date in LastFridaysOfYear(year))
```

C#

Swift Source Code

```
enum GroundType: Int {
    case Grass
    case Rock
    case Water
    case InTheAir
    case Count
}

class Character {

    // MARK: Dealing with fire

    private var isBurning = false
    private var isInvincible = false

    private var fireEmitter: ParticleEmitter! = nil
    private var smokeEmitter: ParticleEmitter! = nil
    private var whiteSmokeEmitter: ParticleEmitter! = nil

    func haltFire() {

            )
```

Swift

Python Source Code

```
    def enabled(self):
        return self._ydl.params.get('cachedir') is not False

    def store(self, section, key, data, dtype='json'):
        assert dtype in ('json',)

        if not self.enabled:
            return

        fn = self._get_cache_fn(section, key, dtype)
        try:

        except Exception:
            tb = traceback.format_exc()
```

Python

ruby example

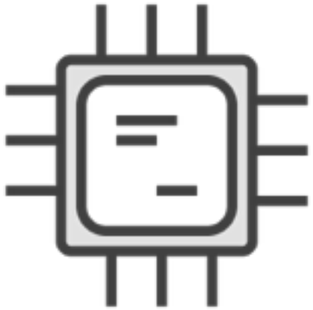
```
class FrontmatterDefaults
  # Initializes a new instance.
  def initialize(site)
    @site = site
  end

  def update_deprecated_types(set)
    return set unless set.key?('scope') && set['scope'].key?('type')

    set['scope']['type'] =
      case set['scope']['type']
      when 'page'

        Deprecator.defaults_deprecate_type('draft', 'drafts')
        'drafts'
      else
```

Ruby

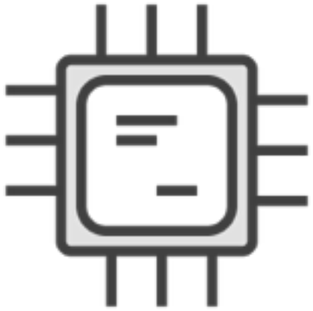
“A computer program is a list of instructions we write using a programming language the computer understands”

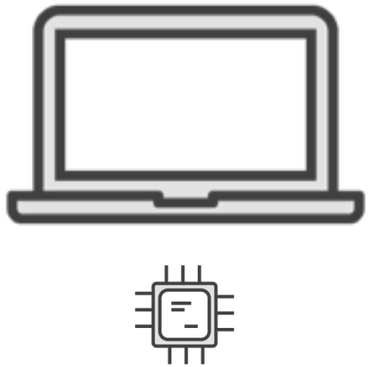
The entire point of
programming
languages
is that we
DON'T
have to write
machine code

The CPU Executes the Program



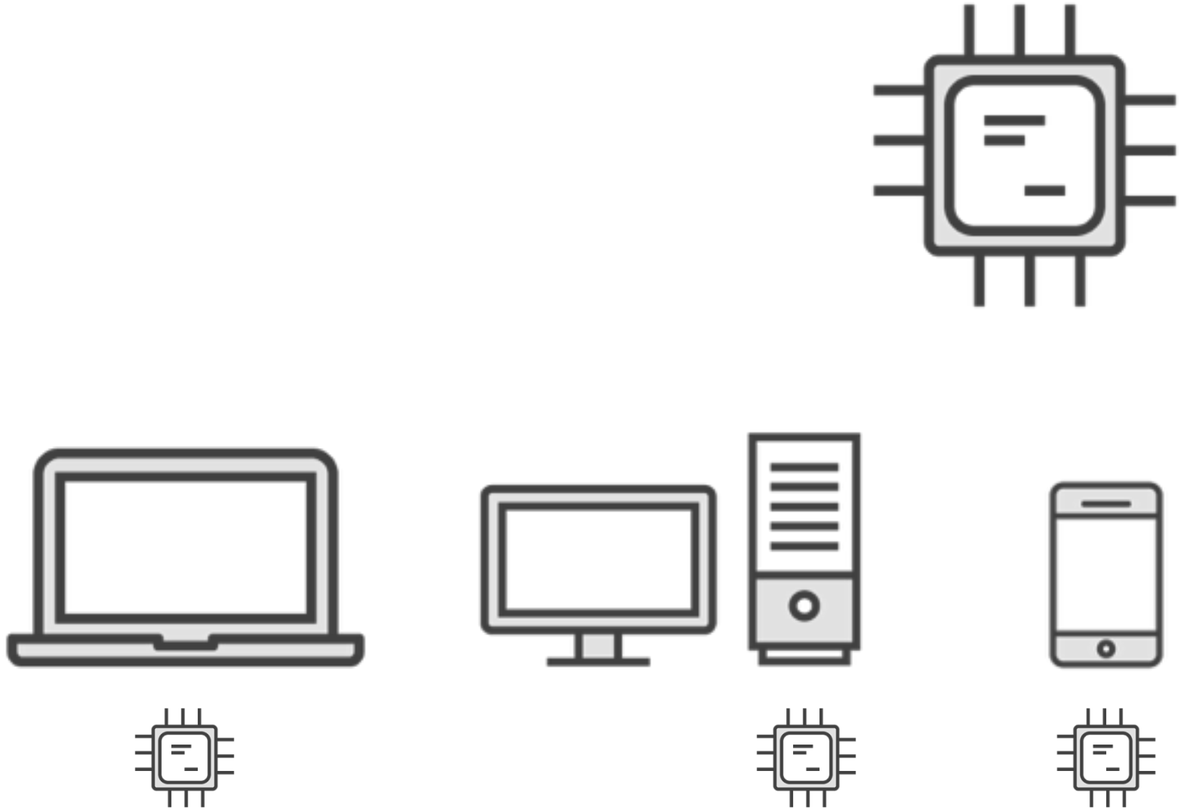
The CPU Executes the Program



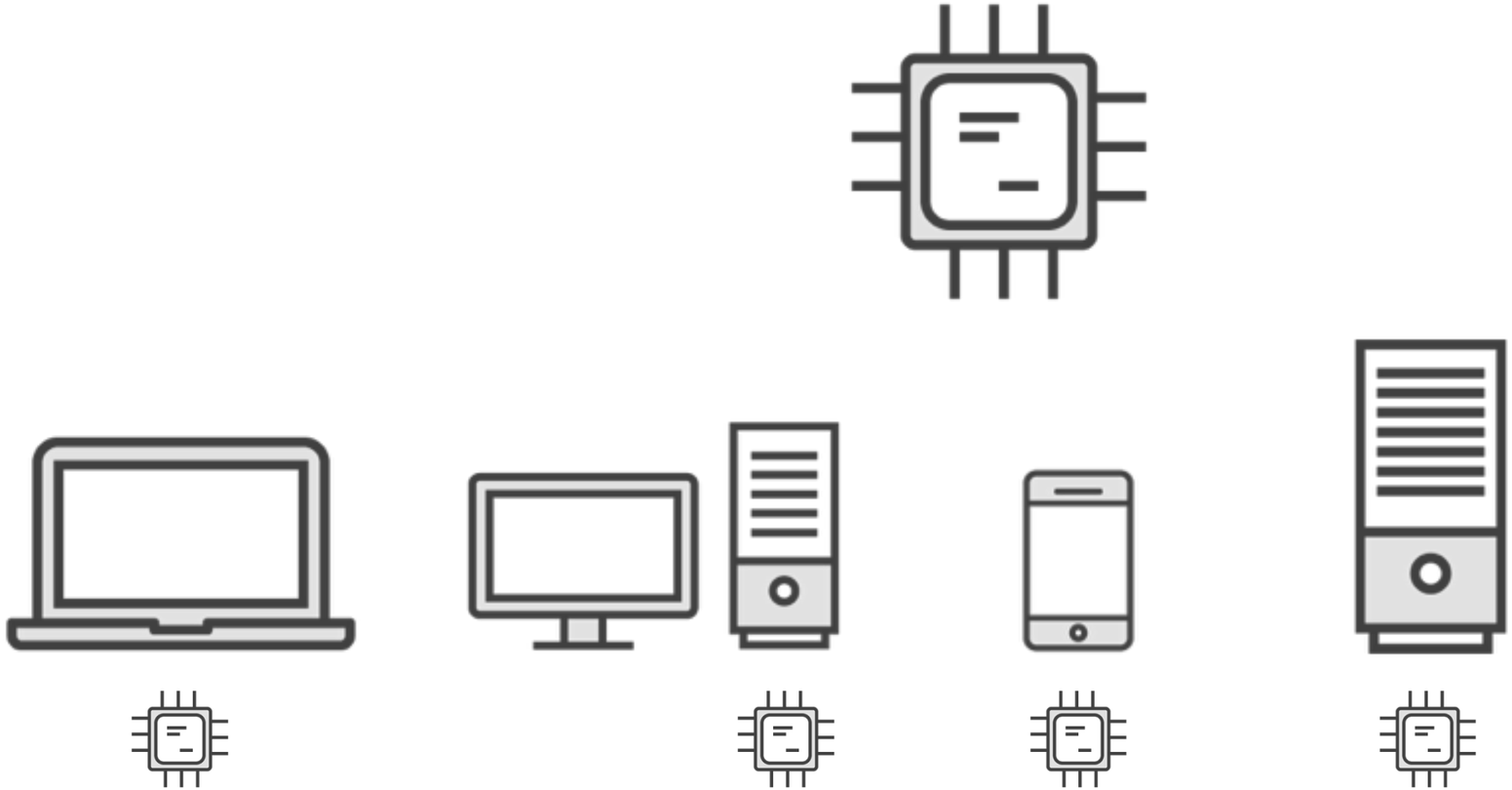


The CPU Executes the Program

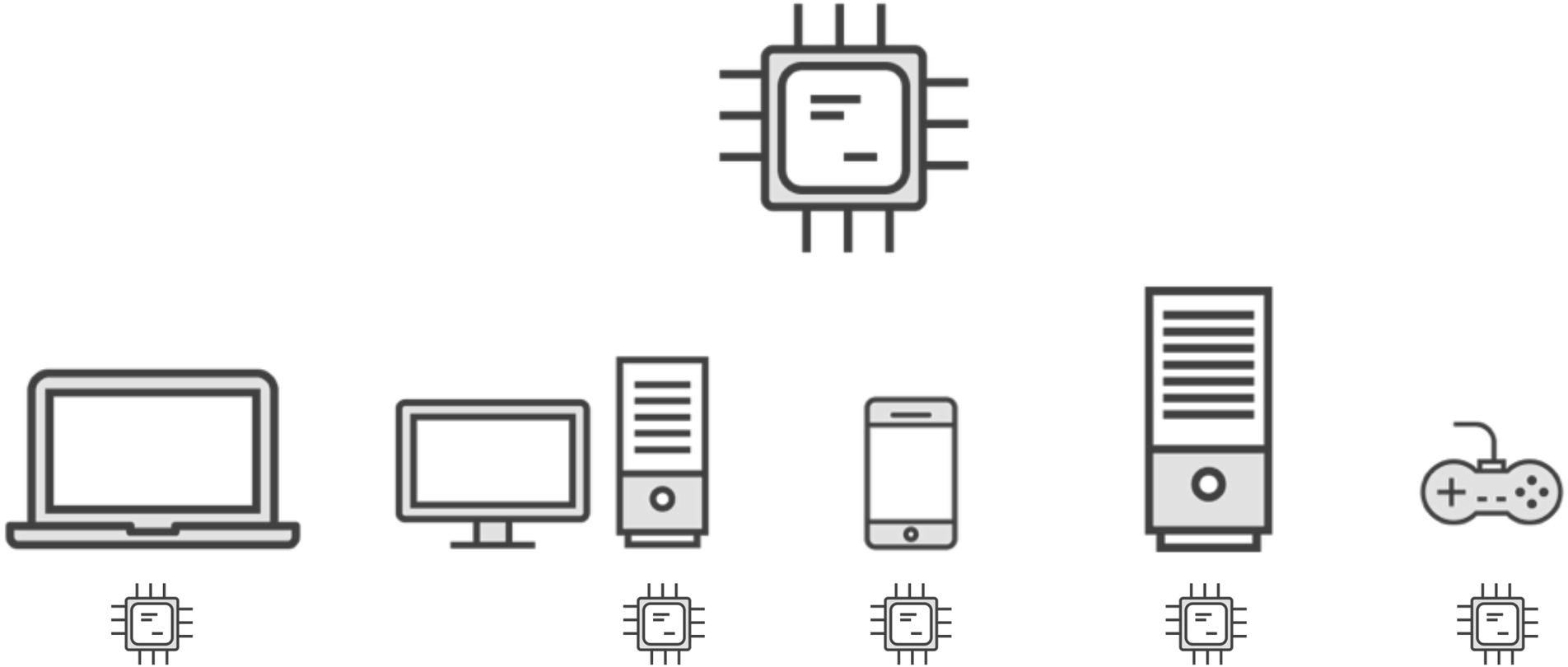
The CPU Executes the Program



The CPU Executes the Program



The CPU Executes the Program



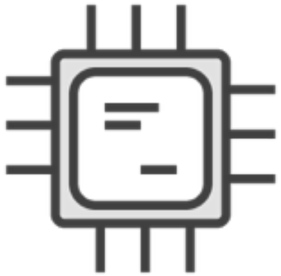
Getting Your Code to Run

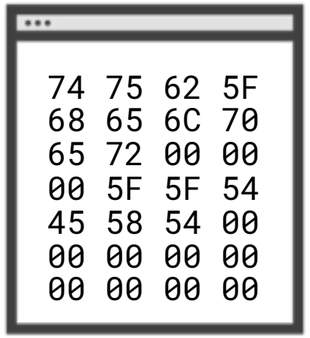
Getting Your Code to Run

Programmers write source code

Getting Your Code to Run

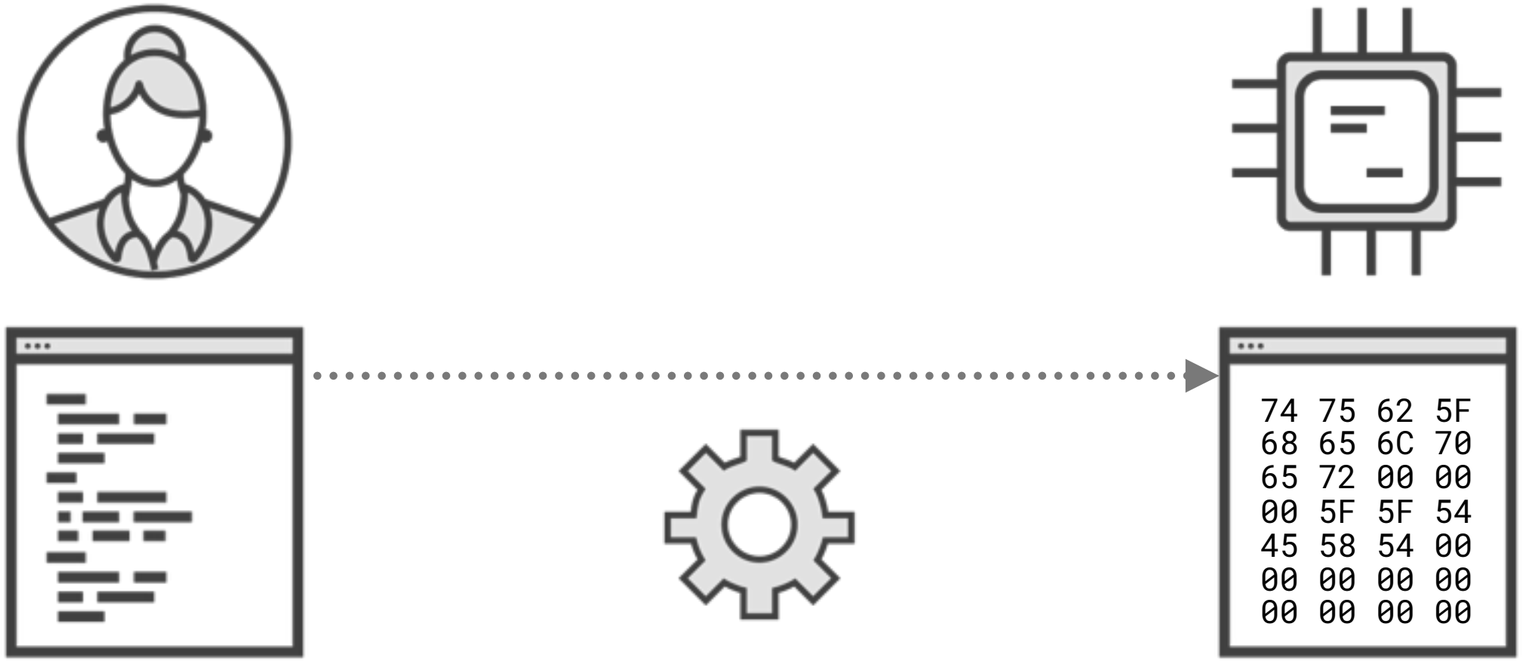
Programmers write source code





CPU understands machine code

Getting Your Code to Run

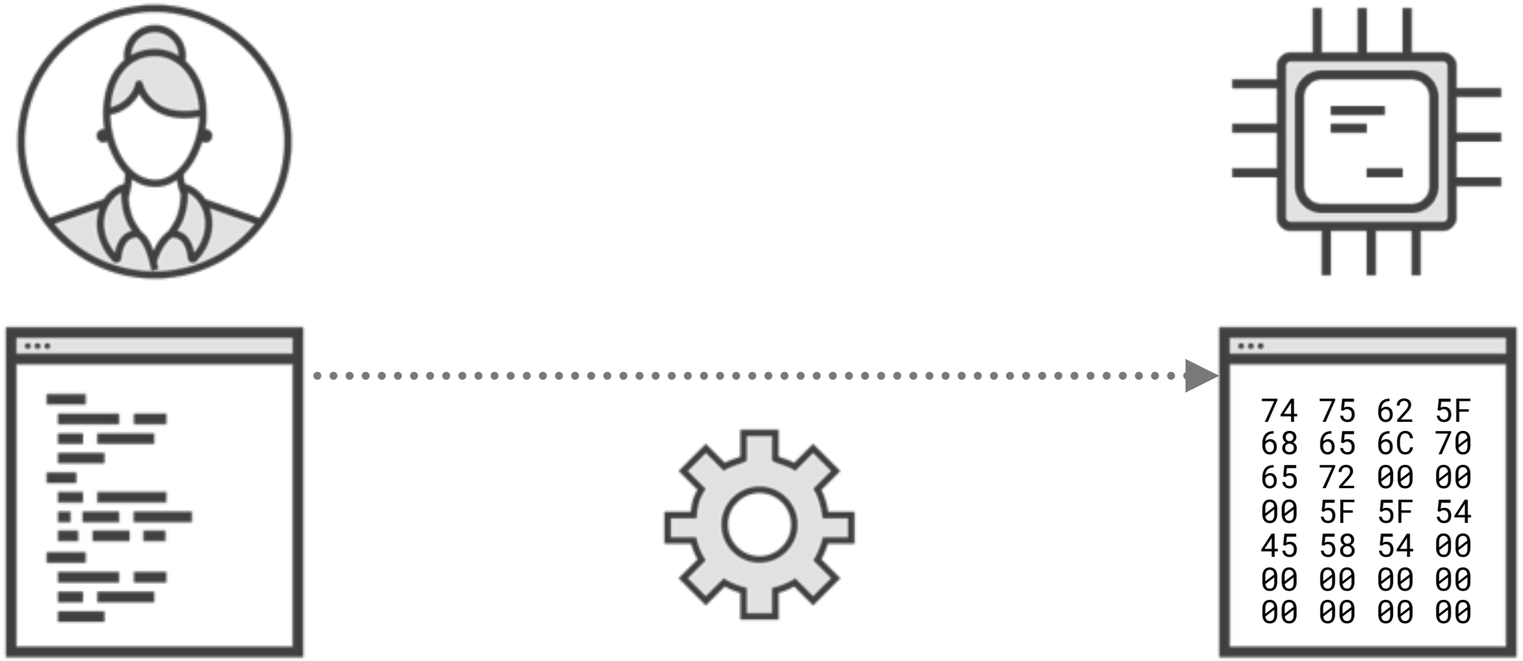


Programmers write source code

conversion process?

CPU understands machine code

Getting Your Code to Run



Programmers write source code

Compile or Interpret

CPU understands machine code

Converting Source Code to Machine Code

Converting Source Code to Machine Code

My computer

Converting Source Code to Machine Code

My computer

Your computer

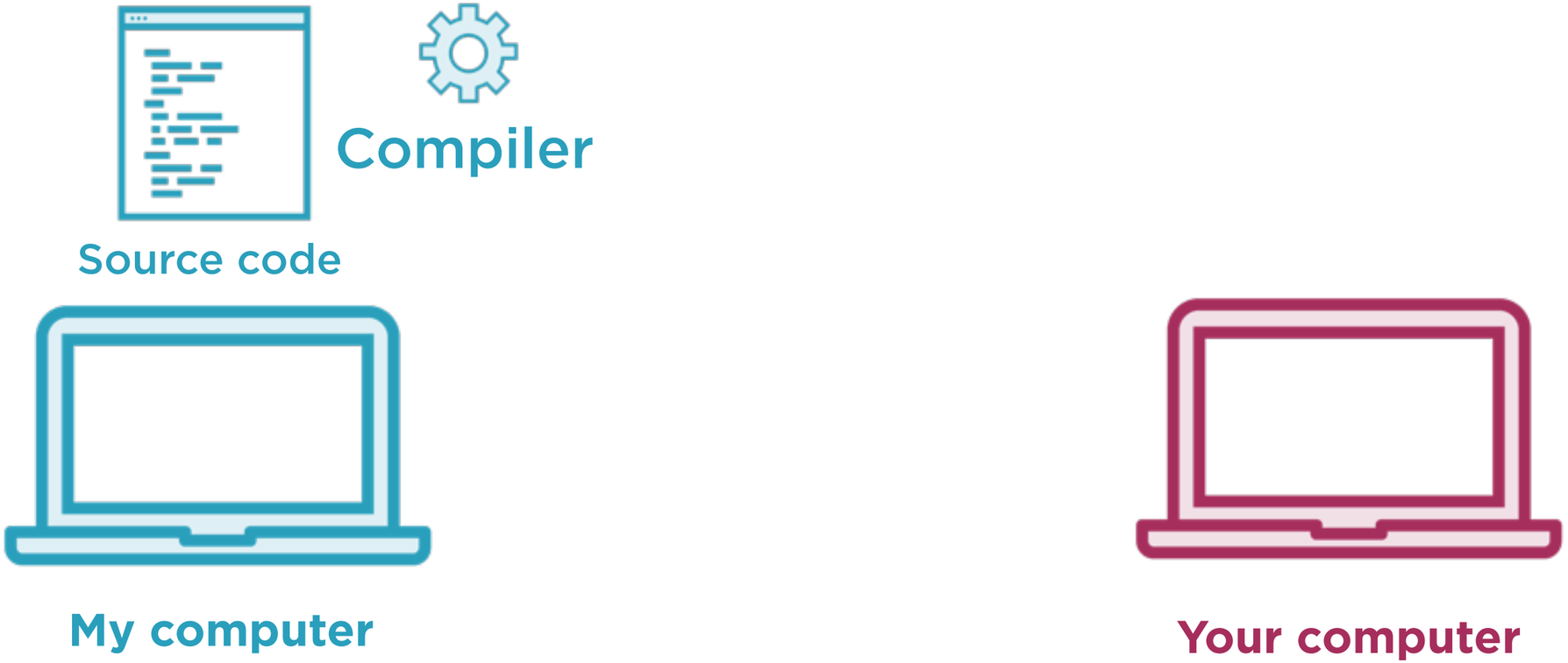
Converting Source Code to Machine Code

Converting Source Code to Machine Code

Option 1: **Compile**

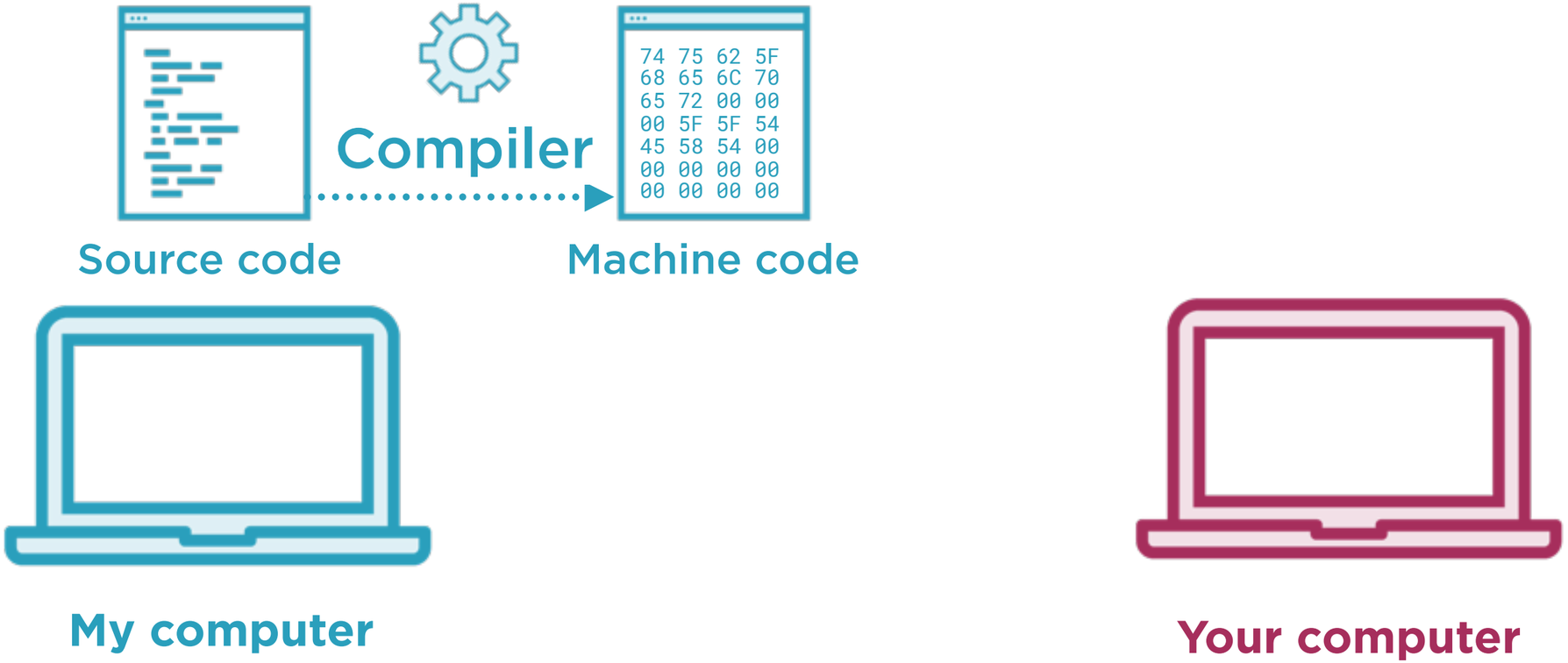
Converting Source Code to Machine Code

Option 1: **Compile**



Converting Source Code to Machine Code

Option 1: **Compile**



Converting Source Code to Machine Code

Option 1: **Compile**

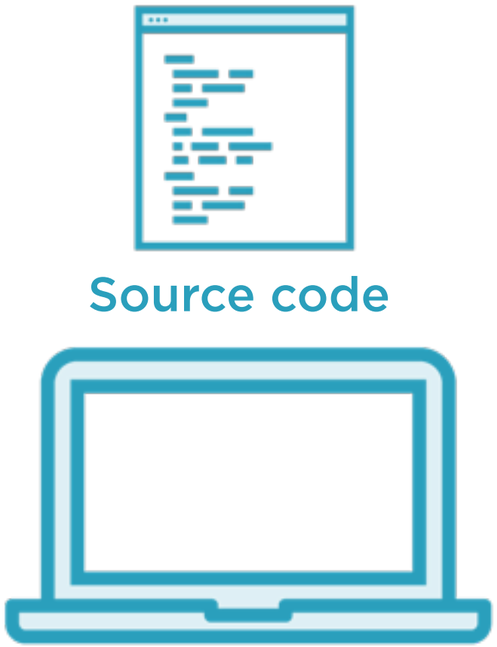
Converting Source Code to Machine Code

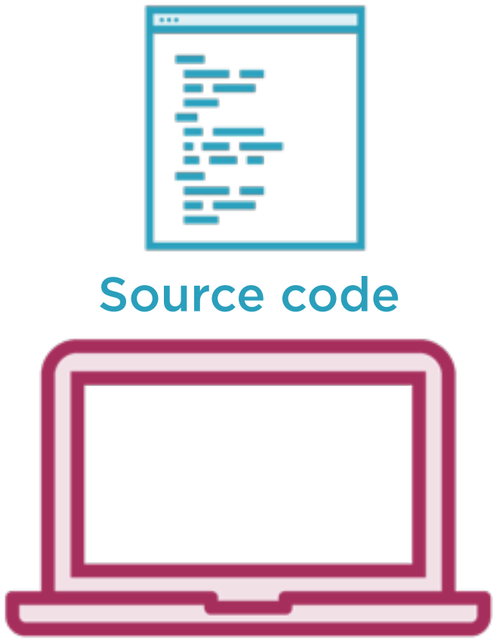
Converting Source Code to Machine Code

Option 2: **Interpret**

Converting Source Code to Machine Code

Option 2: **Interpret**



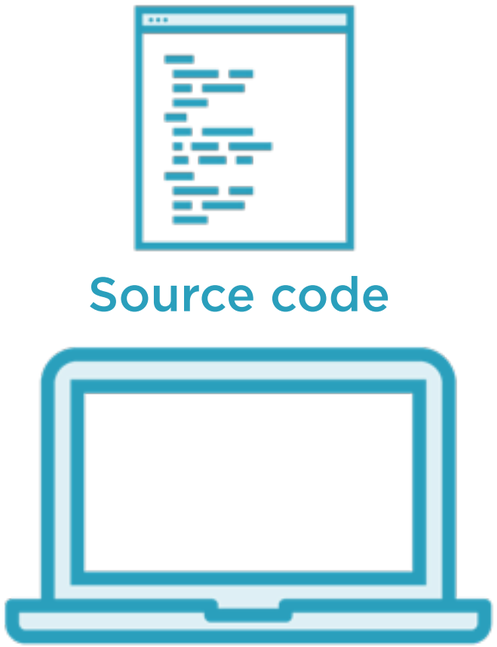


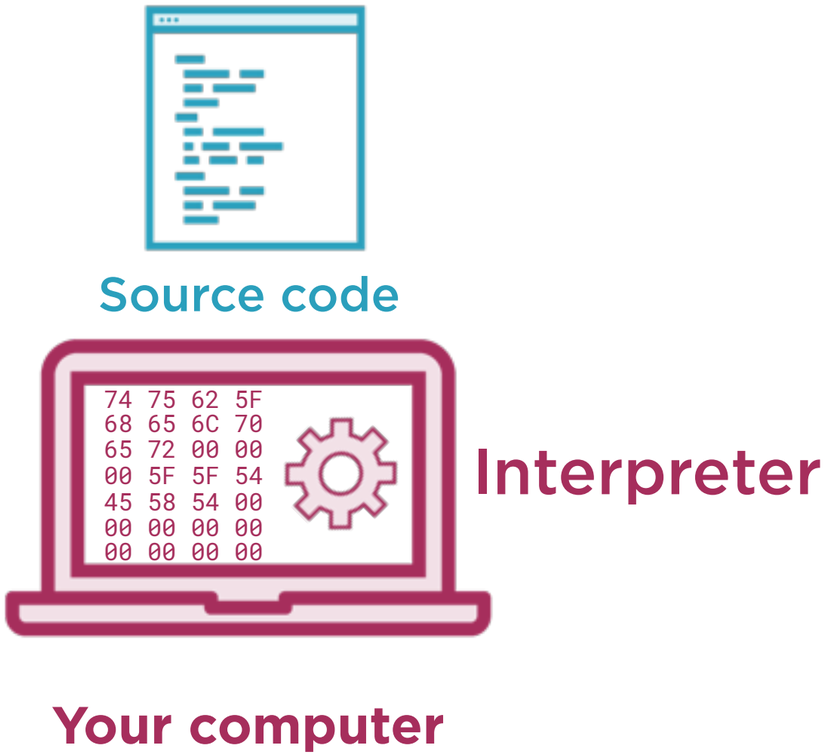
Converting Source Code to Machine Code

Option 2: **Interpret**

Converting Source Code to Machine Code

Option 2: **Interpret**





Pros and Cons

Compiled languages	Interpreted languages
Often faster ✓	✗ Often slower
Platform-specific ✗	✓ Cross-platform
Source code not shared ✗ ✓	✗ ✓ Source code is shared
"Ready to run" machine code ✓	✗ Interpreter required on each computer

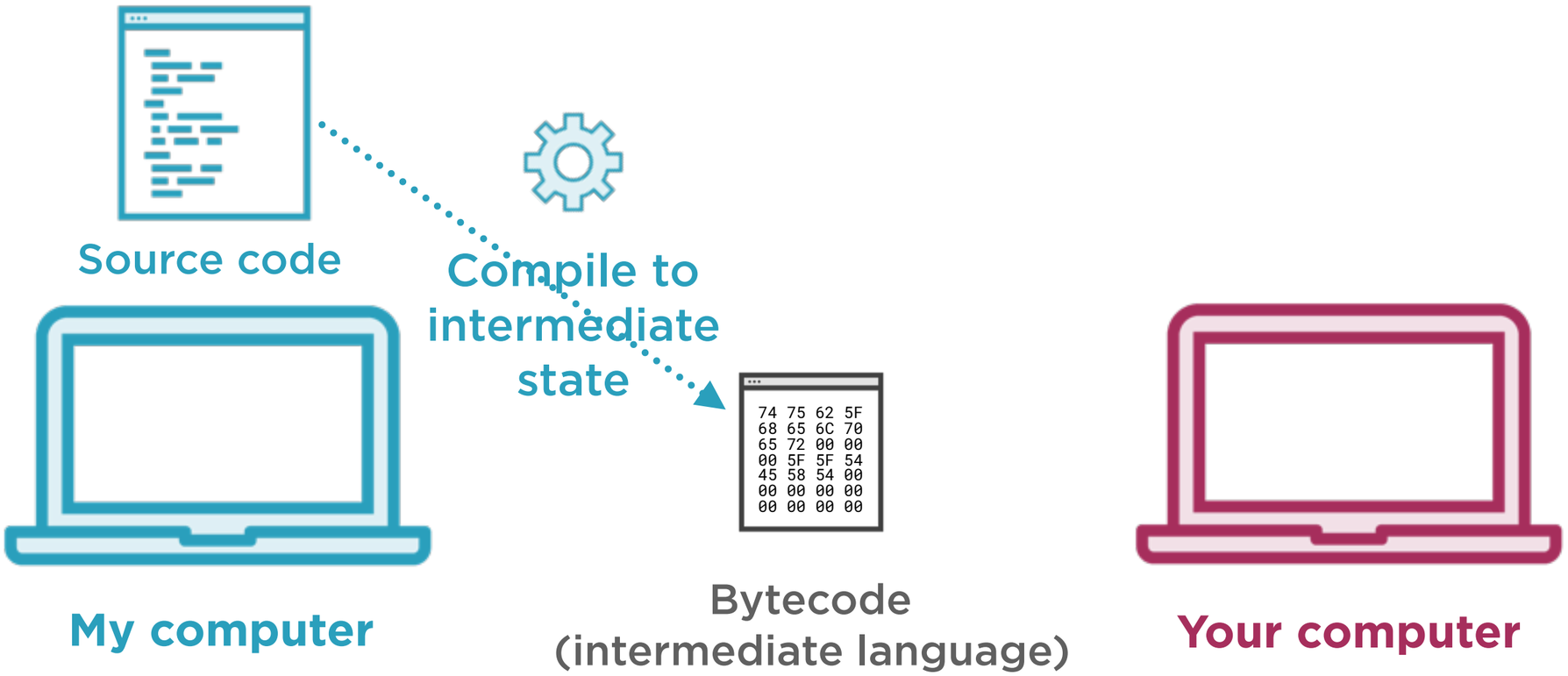
Converting Source Code to Machine Code

Converting Source Code to Machine Code

Option 3: **Intermediate**

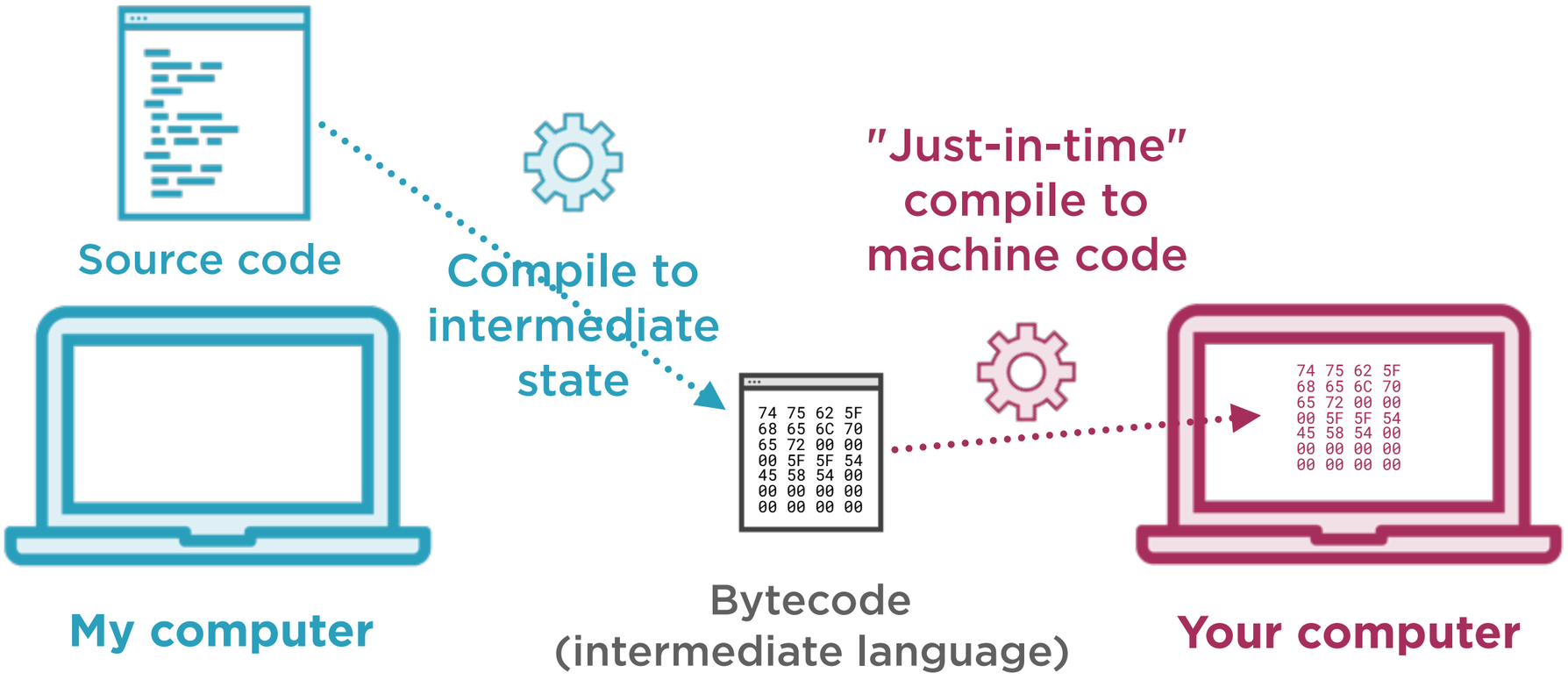
Converting Source Code to Machine Code

Option 3: **Intermediate**



Converting Source Code to Machine Code

Option 3: **Intermediate**



Typical Compilation Models

Typical Compilation Models

Compiled languages

C++
Swift
C

Typical Compilation Models

Compiled languages

C++
Swift
C

Interpreted languages

JavaScript
Python
PHP

Typical Compilation Models

Compiled languages

C++
Swift
C

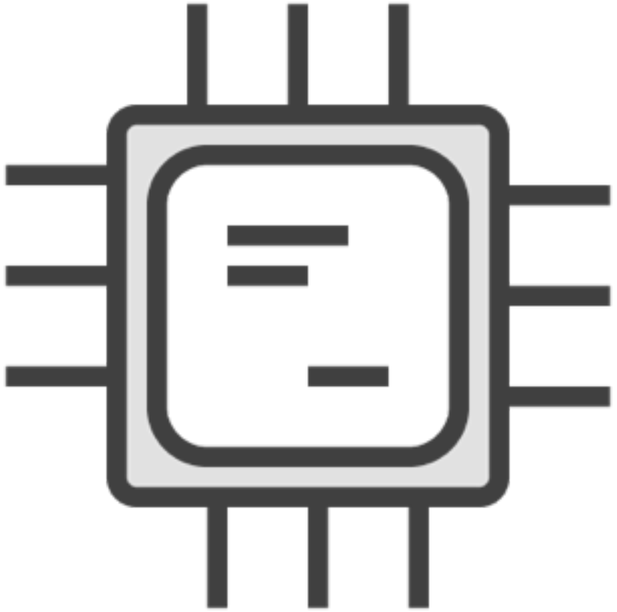
Interpreted languages

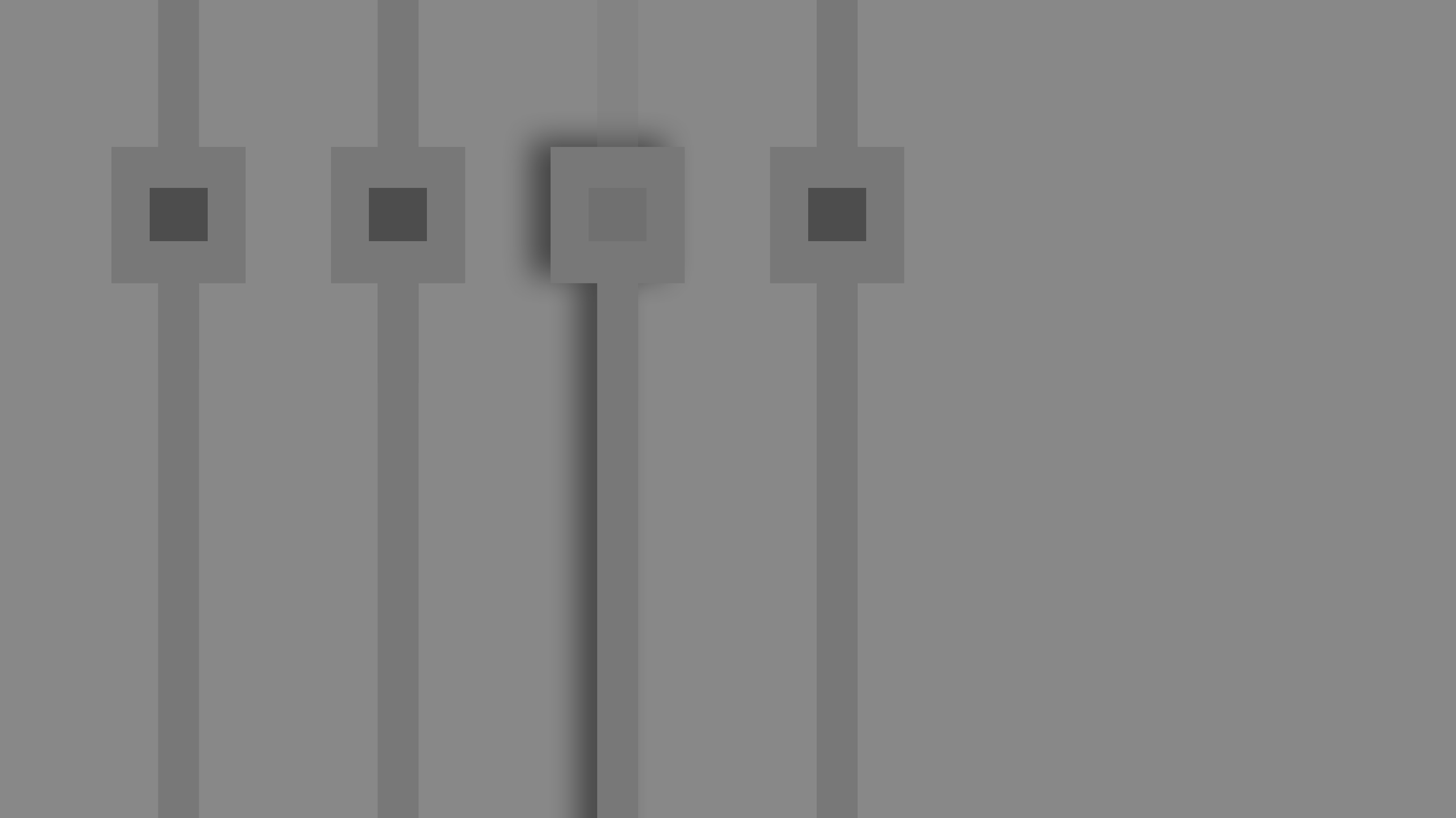
JavaScript
Python
PHP

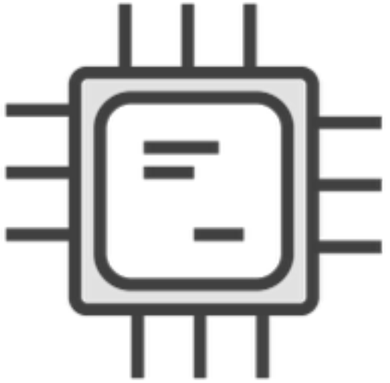
Hybrid languages

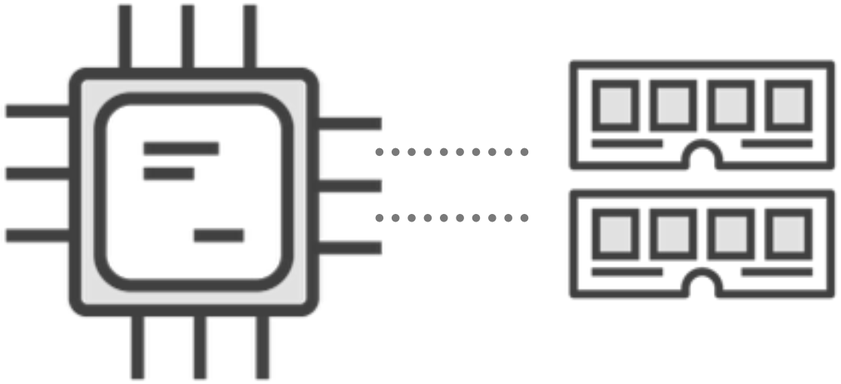
Java
ActionScript
LISP

The CPU









One Switch, Two Values

user is not logged in

One Switch, Two Values

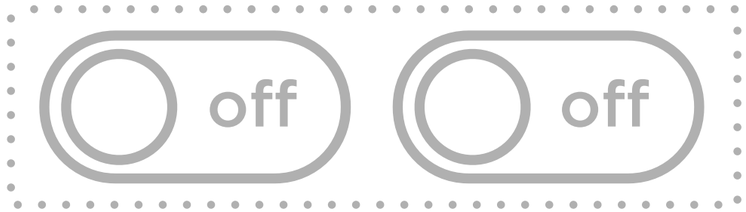
user is not logged in

currently recording

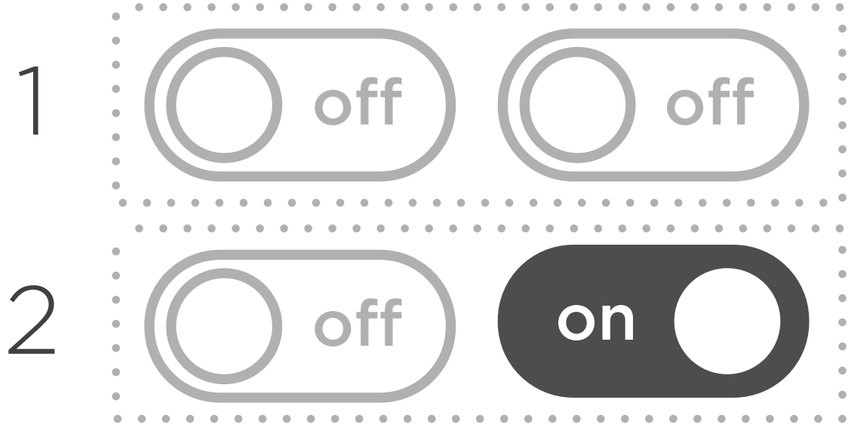
Two Switches, Four Values

Two Switches, Four Values

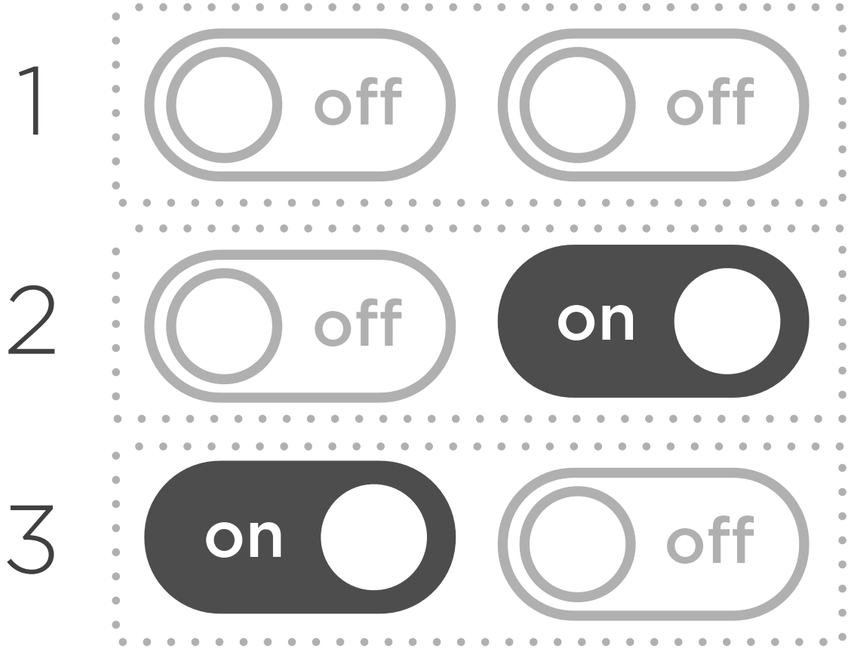
1



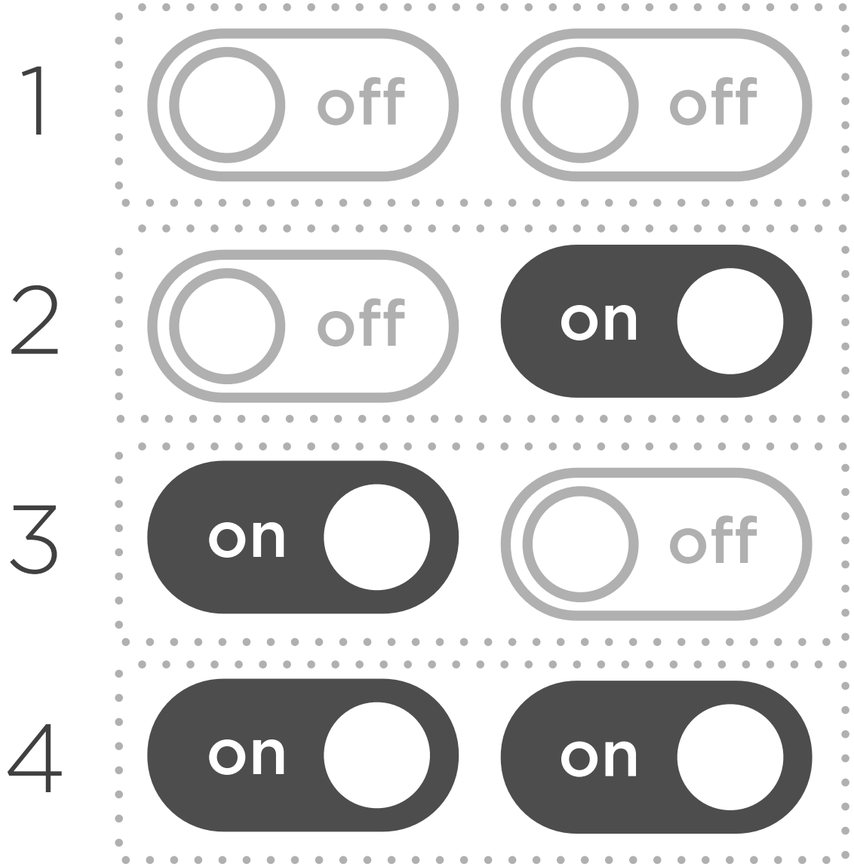
Two Switches, Four Values



Two Switches, Four Values

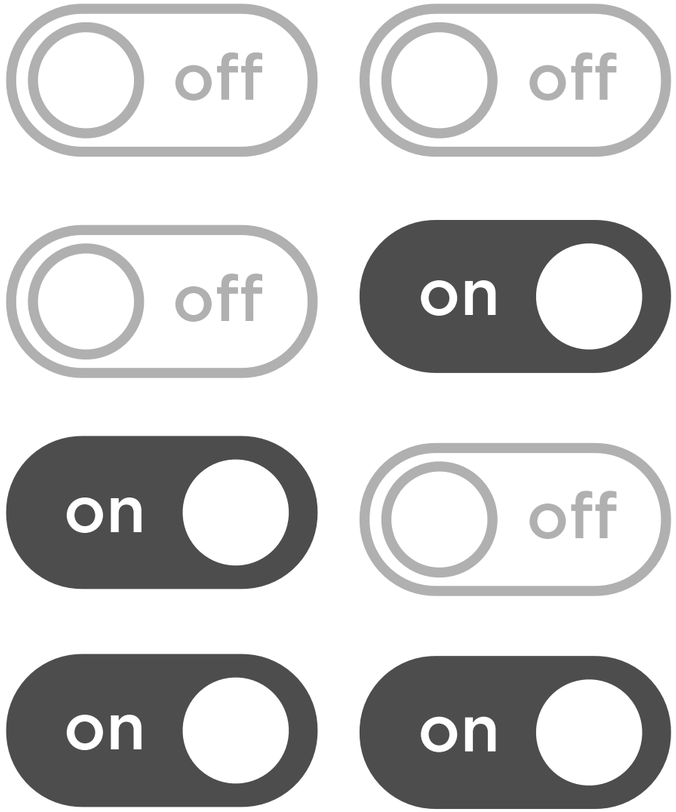


Two Switches, Four Values

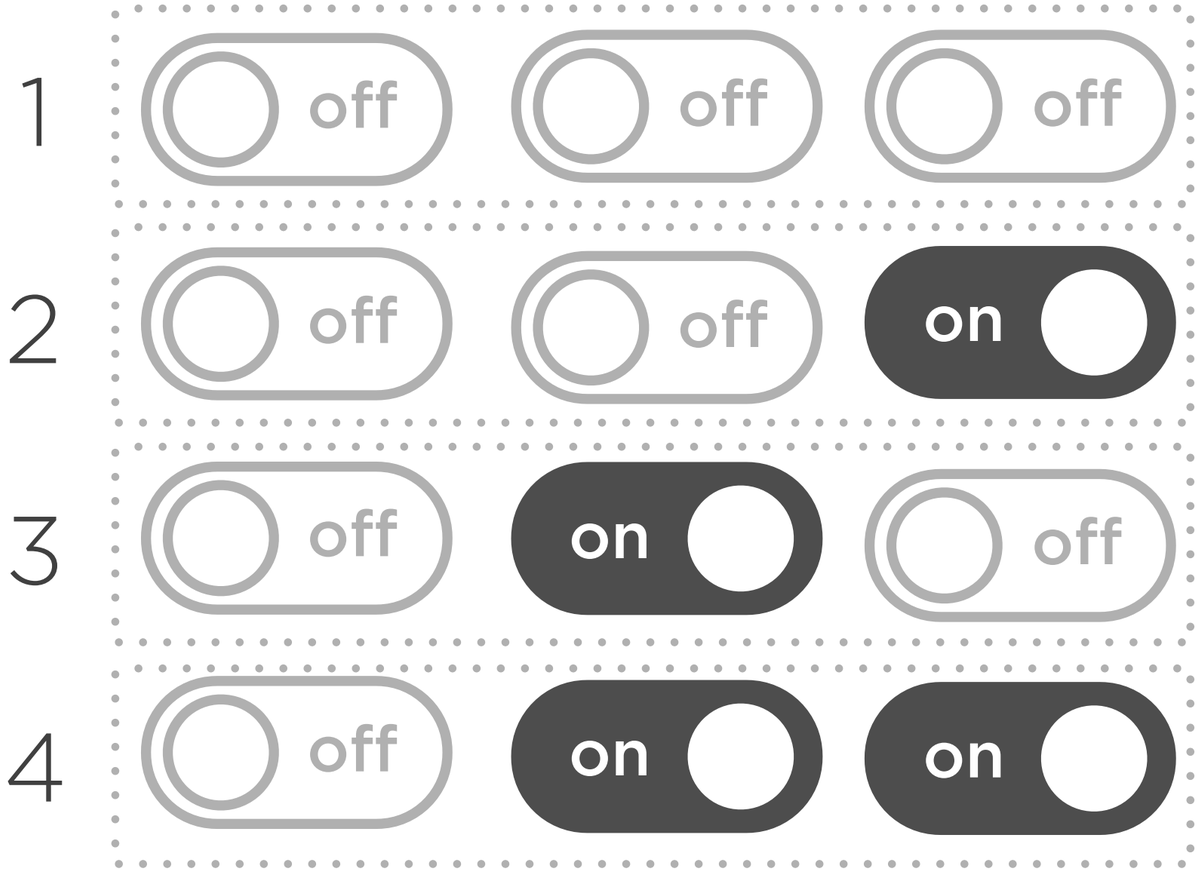


Each Addition Doubles The Options

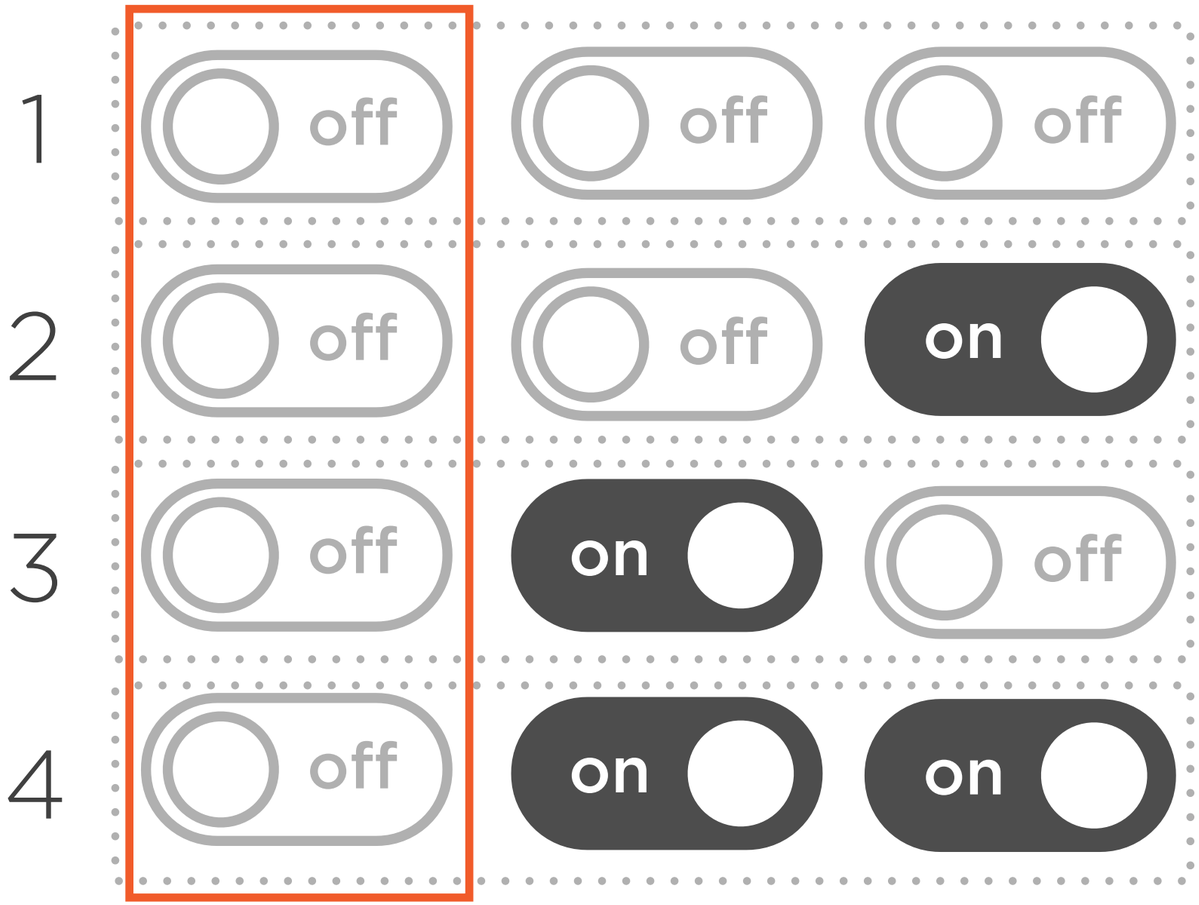
Each Addition Doubles The Options



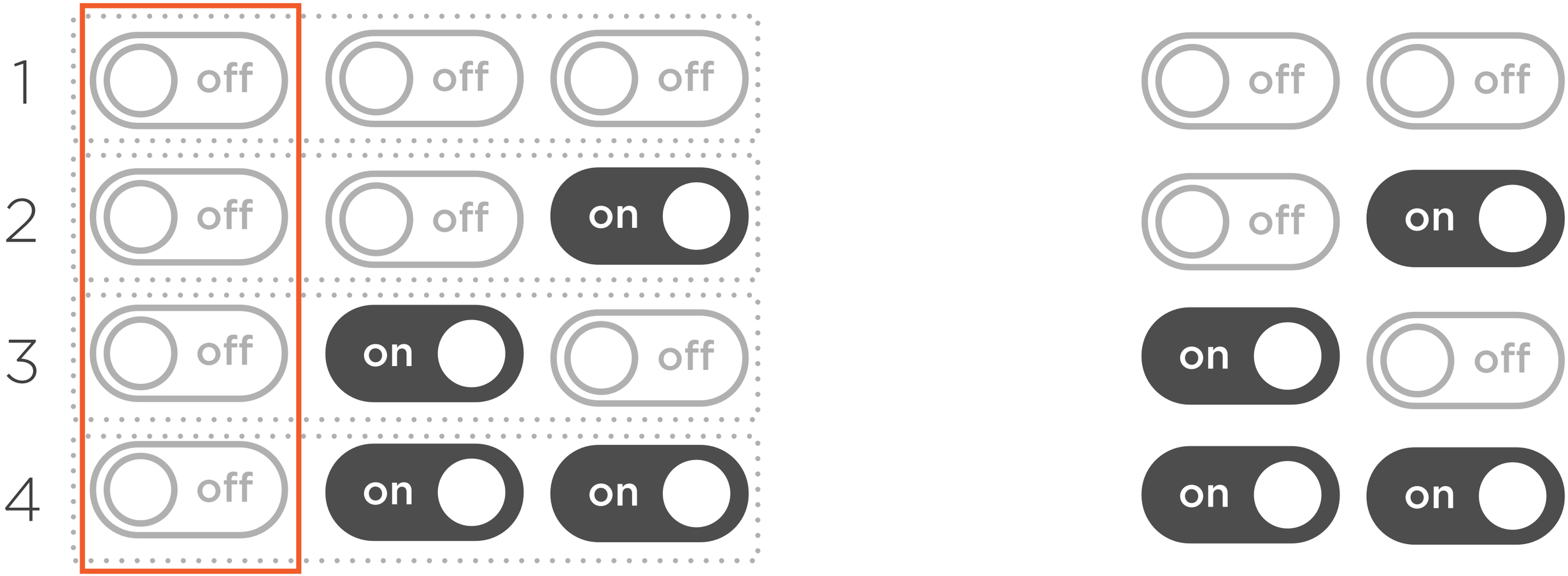
Each Addition Doubles The Options



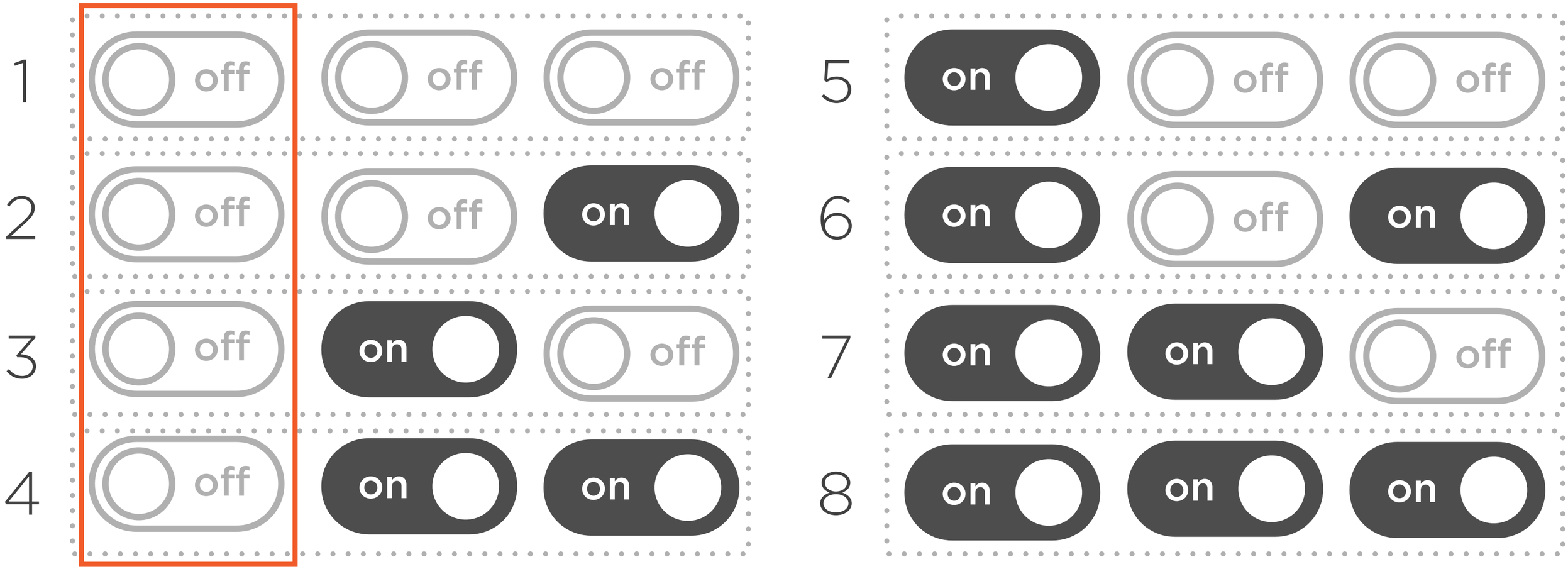
Each Addition Doubles The Options



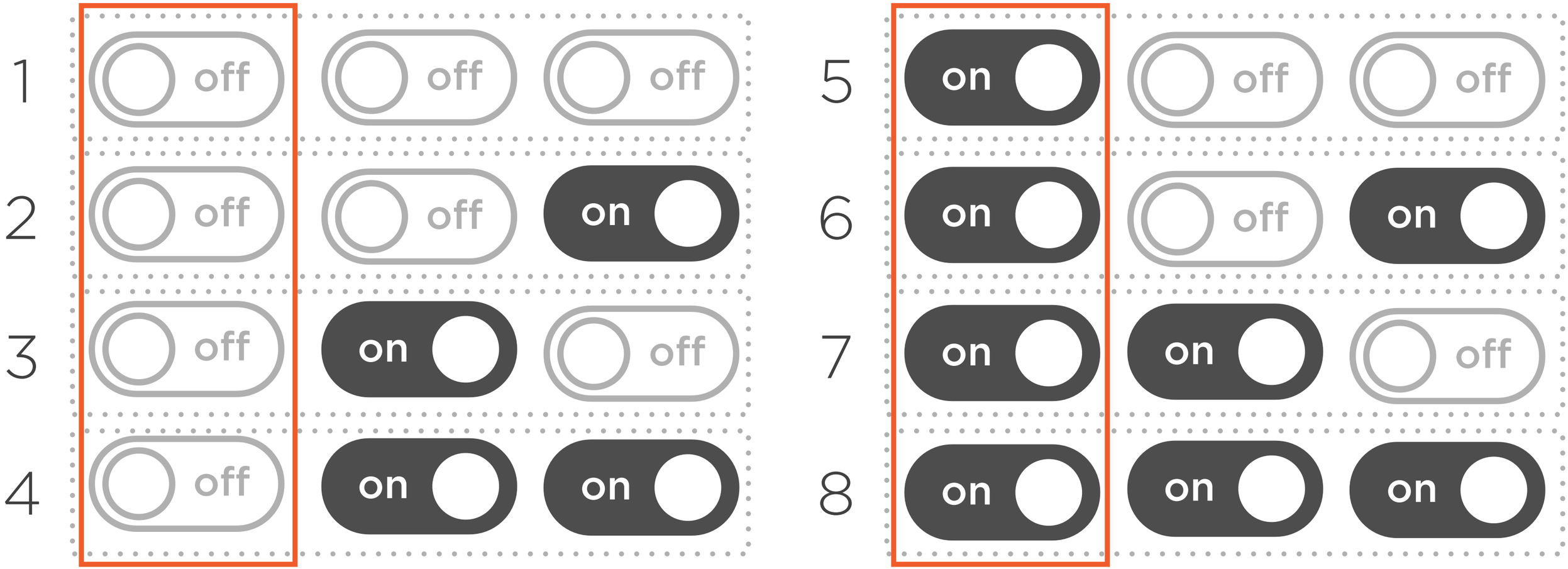
Each Addition Doubles The Options



Each Addition Doubles The Options



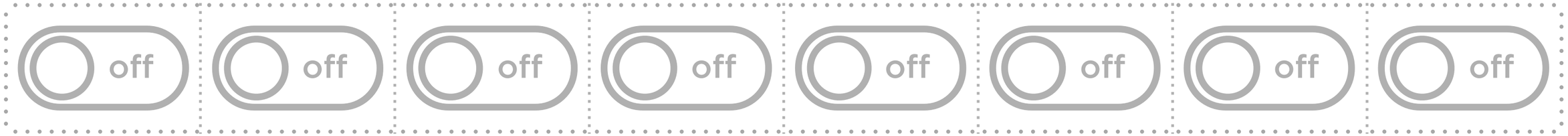
Each Addition Doubles The Options



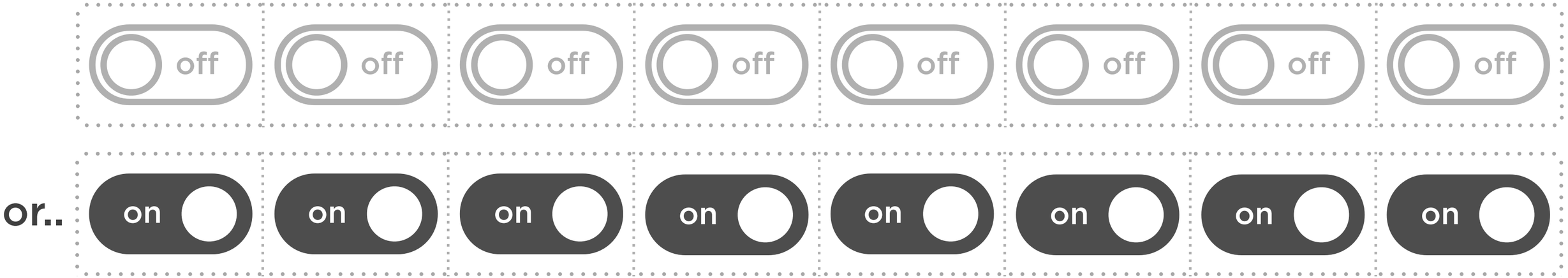
With Four Switches, 16 Arrangements

1	off	off	off	off
2	off	off	off	on
3	off	off	on	off
4	off	off	on	on
5	on	off	off	off
6	on	off	off	on
7	on	off	on	off
8	on	off	on	on
9	off	on	off	off
10	off	on	off	on
11	off	on	on	off
12	off	on	on	on
13	on	on	off	off
14	on	on	off	on
15	on	on	on	off
16	on	on	on	on

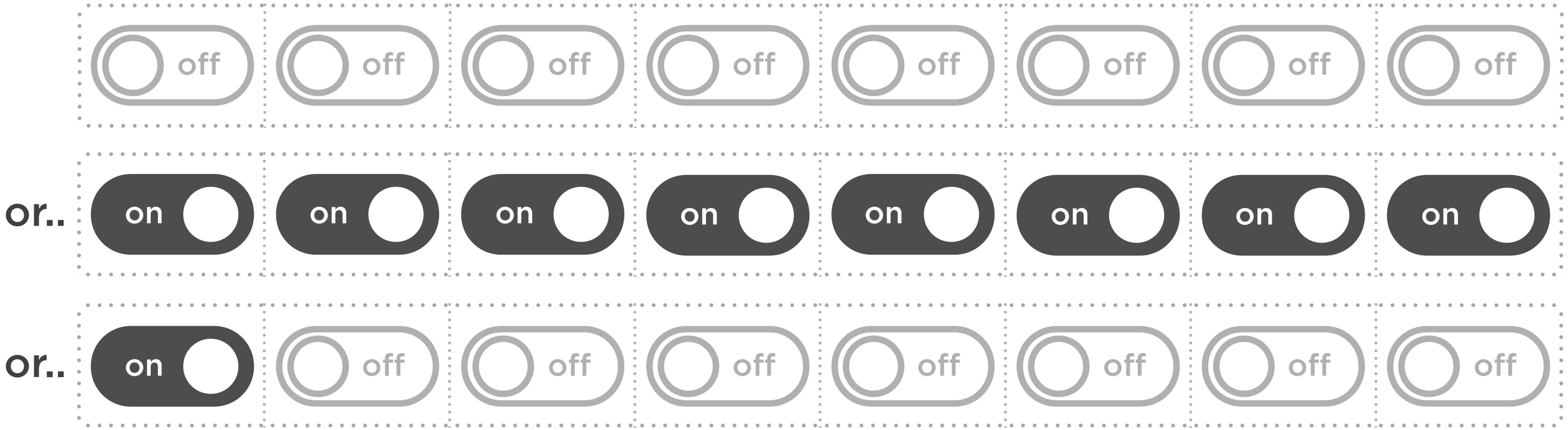
Eight Switches: 256 Arrangements



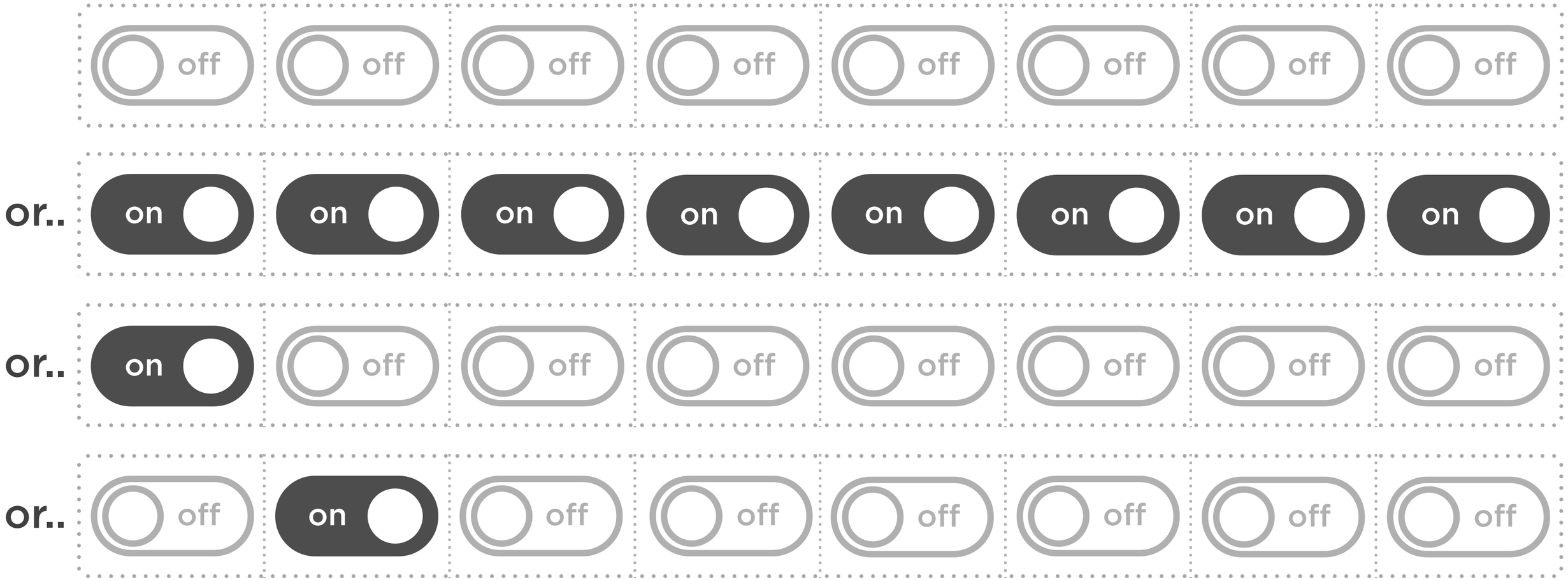
Eight Switches: 256 Arrangements



Eight Switches: 256 Arrangements



Eight Switches: 256 Arrangements



(and 252 more...)

Eight Switches: 256 Arrangements

"Off, Off, Off, Off, Off, Off, Off,Off"

"On, On, On, On, On, On, On, On"

"On, Off, Off, Off, Off, Off, Off, Off"

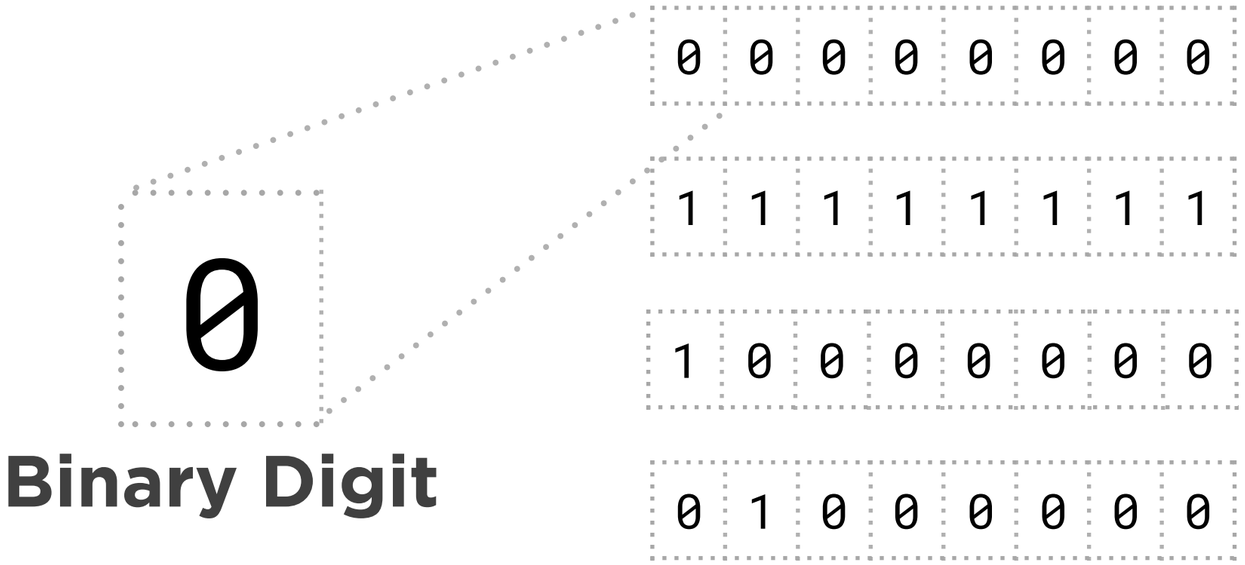
"Off, On, Off, Off, Off, Off, Off, Off"

0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0

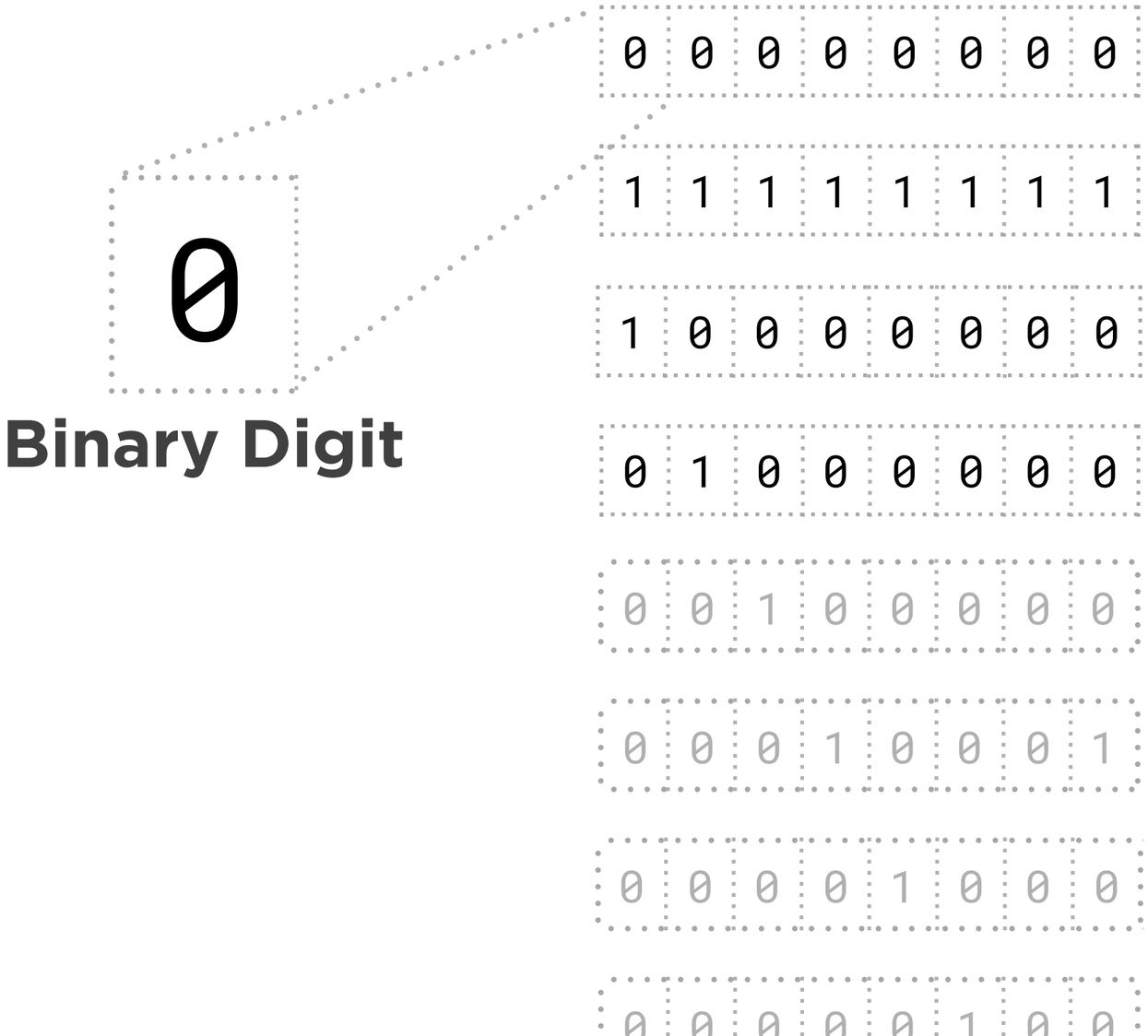
Eight **Bits**: 256 Arrangements

0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0

Eight **Bits**: 256 Arrangements



Eight **Bits**: 256 Arrangements



Need Bigger Values? Use More Bits.

0 0 0 0 0 0 0 0

8 bits: 256 possible values

Need Bigger Values? Use More Bits.

0 0 0 0 0 0 0 0

8 bits: 256 possible values

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

16 bits: 65,000 possible values

Need Bigger Values? Use More Bits.

0 0 0 0 0 0 0 0

8 bits: 256 possible values

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

16 bits: 65,000 possible values

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

32 bits: +4.3 billion possible values

Need Bigger Values? Use More Bits.

0 0 0 0 0 0 0 0

8 bits: 256 possible values

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

16 bits: 65,000 possible values

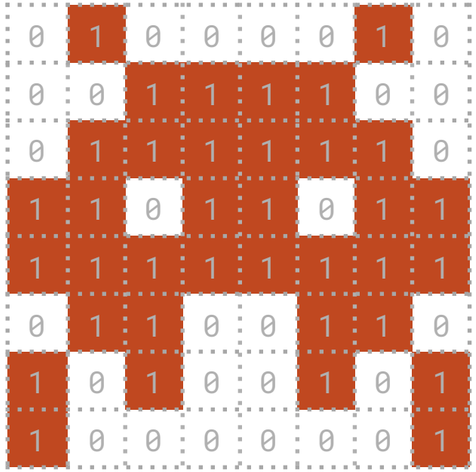
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

32 bits: +4.3 billion possible values

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

64 bits: 18446744073709551616 possible values

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


0 1 0 0 0 0 1 0
0 0 1 1 1 1 0 0
0 1 1 1 1 1 1 0
1 1 0 1 1 0 1 1
1 1 1 1 1 1 1 1
0 1 1 0 0 1 1 0
1 0 1 0 0 1 0 1
1 0 0 0 0 0 0 1

Twenty seven

Twenty seven

27

Twenty seven

27

XXVII

Twenty seven

27

XXVII

Twenty seven

27

XXVII

Binary to Hexadecimal

0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0

Binary to Hexadecimal

0	0	0	0	0	0	0	0	**in binary, is**	0	0	**in hex**
1	1	1	1	1	1	1	1	**in binary, is**	F	F	**in hex**
1	0	0	0	0	0	0	0	**in binary, is**	8	0	**in hex**
0	1	0	0	0	0	0	0	**in binary, is**	4	0	**in hex**