



*Traverse the Ol'*

**CONDITIONAL CANYON**



**LEVEL 2**

**CONDITIONAL CANYON**

# THE CURRENT BUILD FOR OUR TRAIN STATUS SYSTEM

Our system currently prints both operational and non-operational trains

1



2



3



4



5



6



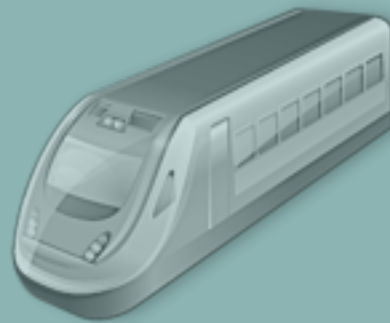
7



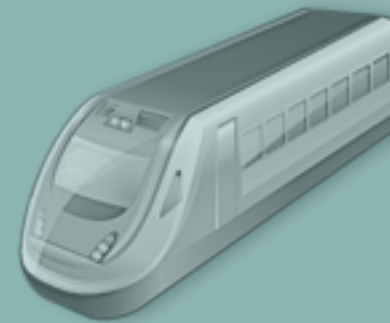
8



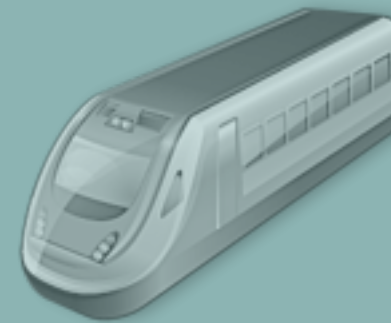
9



10



11



12



# OUR CURRENT SYSTEM USES TWO LOOPS...

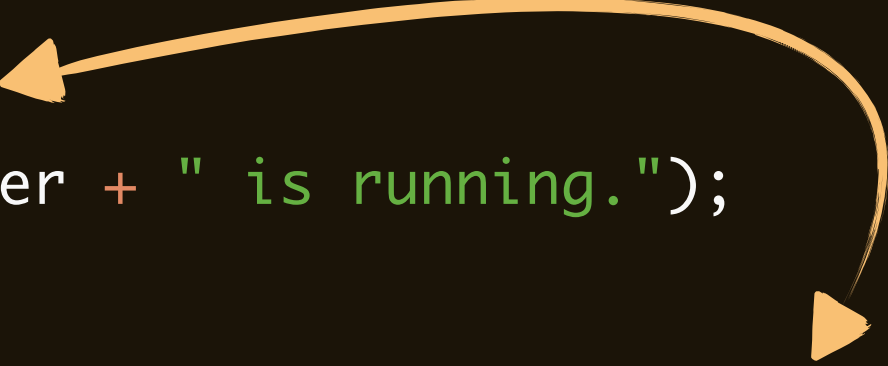
## trains.js

```
var totalTrains = 12;
var trainsOperational = 8;

var trainNumber = 1;
while(trainNumber <= trainsOperational){
    console.log("Train #" + trainNumber + " is running.");
    trainNumber++;
}

for(var stoppedTrain = trainsOperational + 1; stoppedTrain <= totalTrains; stoppedTrain++){
    console.log("Train #" + stoppedTrain + " is not operational.");
}
```

We want to merge these loops into one loop that DECIDES which trains are running and which aren't.

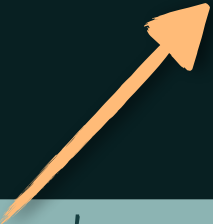


# LETS USE ONE LOOP INSTEAD OF TWO

So how do we run different lines of code based on specific conditions?

trains.js

```
...  
for (var trainNumber = 1; trainNumber <= totalTrains; trainNumber++){  
    *if the train is currently running, we want:*  
    console.log("Train #" + trainNumber + " is running.");  
  
    *otherwise, if it's not, we want:*  
    console.log("Train #" + trainNumber + " is not operational.");  
}
```



Notice now that we're now using only one variable to identify the train's number.



# IF, AND HER BUDDY, ELSE

If and Else allow us to execute certain code based on specific conditions

If the conditional evaluates to true, the code block is executed.

```
if (*some condition is true*) {  
    *do this code!*  
}  
else {  
    *OTHERWISE, do this code instead!*  
}
```

Else follows up with code to execute ONLY when the If conditional is not satisfied. It is ignored otherwise.

# IF, AND HER BUDDY, ELSE

A basic example of conditional execution

```
var value1 = 4;  
var value2 = 9;  
if ( value1 < value2 ) {  
    console.log(value1 + " is less than " + value2);  
} else {  
    console.log(value1 + " is greater than or equal to " + value2);  
}
```

We aren't sure whether it's strictly greater than, only that it is not less than.

→ 4 is less than 9

# IF, AND HER BUDDY, ELSE

## A basic example of conditional execution

```
var value1 = 12;  
var value2 = 9;  
if ( value1 < value2 ) {  
    console.log(value1 + " is less than " + value2);  
} else {  
    console.log(value1 + " is greater than or equal to " + value2);  
}
```

Now, this conditional will evaluate to false, and so the 'else' block will trigger.

→ 12 is greater than or equal to 9

Trust us on this.



# CAN IF AND ELSE MAKE OUR TRAINS.JS BETTER?

## Using conditionals for efficiency

### trains.js

```
var totalTrains = 12;
var trainsOperational = 8;

var trainNumber = 1;
while (trainNumber <= trainsOperational){
    console.log("Train #" + trainNumber + " is running.");
    trainNumber++;
}
for (var stoppedTrain = trainsOperational + 1; stoppedTrain <= totalTrains; stoppedTrain++) {
    console.log("Train #" + stoppedTrain + " is not operational.");
}
```



# BUILDING OUR NEW SYSTEM STATUS LOOP

## Looping with If and Else controls

trains.js


```
...  
for (var trainNumber = 1; trainNumber <= totalTrains; trainNumber++){  
    *if the train is currently running, we want:*  
    console.log("Train #" + trainNumber + " is running.");  
  
    *otherwise, if it's not, we want:*  
    console.log("Train #" + trainNumber + " is not operational.");  
}
```




# BUILDING OUR NEW SYSTEM STATUS LOOP

## Looping with If and Else controls

trains.js
















```
...  
for (var trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
    if (trainNumber <= operationalTrains) {  
        console.log("Train #" + trainNumber + " is running.");  
    } else {  
        console.log("Train #" + trainNumber + " is not operational.");  
    }  
}
```



As soon as trainNumber is no longer within the amount of operational trains, the If block is skipped and the Else block begins printing in each new loop.

# RUNNING OUR NEW SINGLE LOOP!



trainNumber	LOOP: trainNumber<=12?	Is trainNumber<=8?	OUTPUT
 1	TRUE	YES -> IF	Train #1 is running.
 2	TRUE	YES -> IF	Train #2 is running.
 3	TRUE	YES -> IF	Train #3 is running.
 4	TRUE	YES -> IF	Train #4 is running.
 5	TRUE	YES -> IF	Train #5 is running.
 6	TRUE	YES -> IF	Train #6 is running.
 7	TRUE	YES -> IF	Train #7 is running.
 8	TRUE	YES -> IF	Train #8 is running.
 9	TRUE	NO -> ELSE	Train #9 is not operational.
 10	TRUE	NO -> ELSE	Train #10 is not operational.
 11	TRUE	NO -> ELSE	Train #11 is not operational.
 12	TRUE	NO -> ELSE	Train #12 is not operational.
 13	FALSE	STOP THE LOOP!	

# ADDING A SPECIAL TRAIN THAT STARTS LATER

Let's add a train that isn't operational yet, but starts at noon.

1



2



3



4



5



6



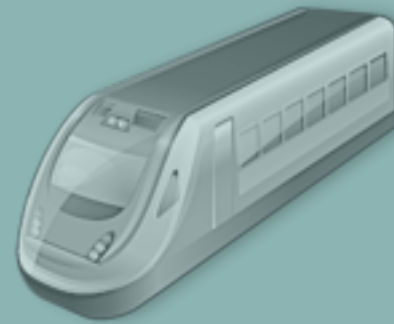
7



8



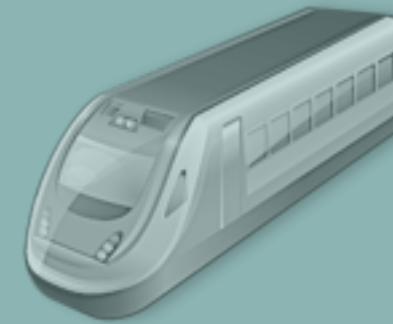
9



10



11



12



# THE ELSE-IF SYNTAX

When two conditions just isn't enough!

```
if (*some condition is true*) {  
  
    *do this code!*  
  
} else if (*some OTHER condition is true*) {  
  
    *do something for this condition!*  
  
} else {  
  
    *IN ALL OTHER CASES, do this code instead!*  
  
}
```

Remember that as soon as a condition is met in any block, the rest will be skipped entirely!



# CHECKING MULTIPLE CONDITIONS

“Else If” can be used when many specific scenarios need attention

```
if (trainNumber <= operationalTrains) {  
    console.log("Train #" + trainNumber + " is running.");  
} *otherwise, first check if the train is the express train*  
    console.log("Train #10 begins running at noon.");  
} else {  
    console.log("Train #" + trainNumber + " is not operational.");  
}
```

# CHECKING MULTIPLE CONDITIONS

“Else If” can be used when many specific scenarios need attention

```
if (trainNumber <= operationalTrains) {
```

```
    console.log("Train #" + trainNumber + " is running.");
```

```
} else if (trainNumber == 10) {
```

This condition is checked ONLY when a train is NOT an operational train. Thus, train 10 only starts at noon if it is not ALREADY an operational train.

```
    console.log("Train #10 begins running at noon.");
```

```
} else {
```

Triggers only when a train is neither regular NOR express

```
    console.log("Train #" + trainNumber + " is not operational.");
```

```
}
```



# UPDATING OUR SYSTEM STATUS LOOP














Now we can print based on multiple conditions!

trains.js

```
...
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
    if (trainNumber <= operationalTrains) {
        console.log("Train #" + trainNumber + " is running.");
    } else if (trainNumber == 10) {
        console.log("Train #10 begins running at noon.");
    } else {
        console.log("Train #" + trainNumber + " is not operational.");
    }
}
```



# SO WHAT DOES THIS GET US?

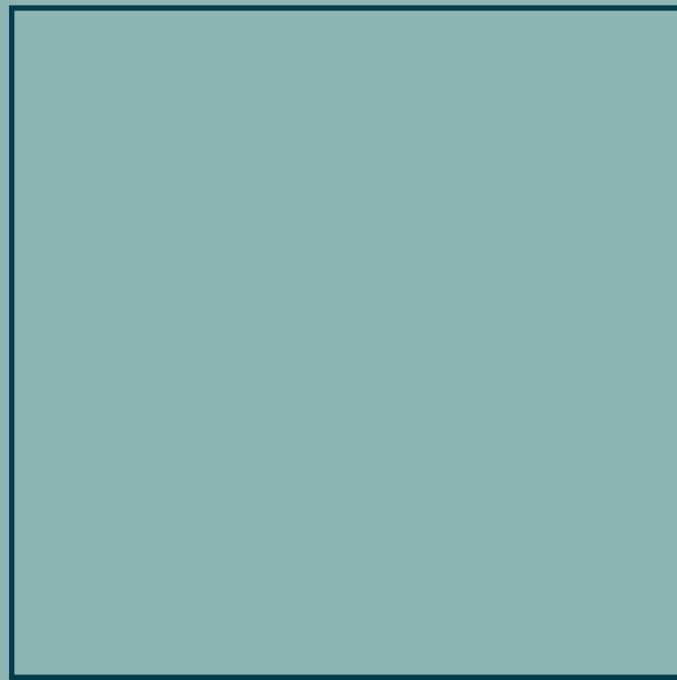
trainNumber	LOOP: trainNumber<=12?	Is trainNumber<=8?	Is trainNumber ==10?	OUTPUT
 1	TRUE	YES -> IF	IGNORE	Train #1 is running.
 2	TRUE	YES -> IF	IGNORE	Train #2 is running.
 3	TRUE	YES -> IF	IGNORE	Train #3 is running.
 4	TRUE	YES -> IF	IGNORE	Train #4 is running.
 5	TRUE	YES -> IF	IGNORE	Train #5 is running.
 6	TRUE	YES -> IF	IGNORE	Train #6 is running.
 7	TRUE	YES -> IF	IGNORE	Train #7 is running.
 8	TRUE	YES -> IF	IGNORE	Train #8 is running.
 9	TRUE	NO	NO -> ELSE	Train #9 is not operational.
 10	TRUE	NO	YES -> ELSE-IF	Train #10 begins running at noon.
 11	TRUE	NO	NO -> ELSE	Train #11 is not operational.
 12	TRUE	NO	NO -> ELSE	Train #12 is not operational.
 13	FALSE	STOP THE LOOP!		

# NESTED CONDITIONALS

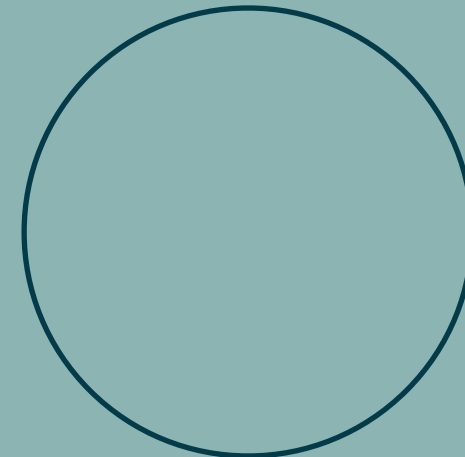
## Splitting results for a single condition

Let's say we had some shapes.

*We have two sizes for squares...*



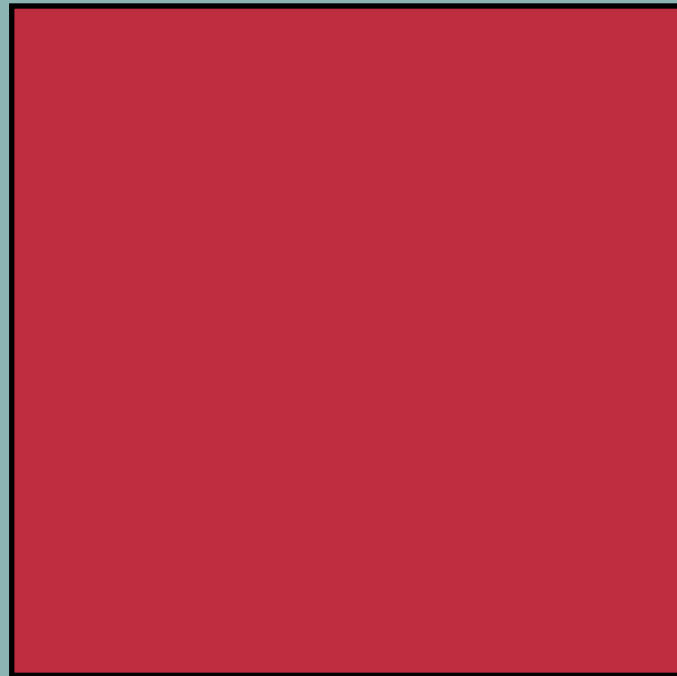
*...but only one size for circles.*



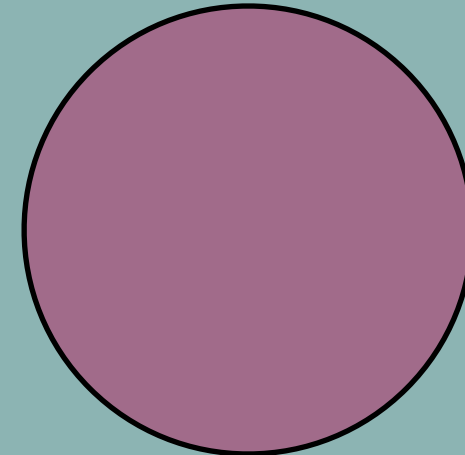
# NESTED CONDITIONALS

## Splitting results for a single condition

We want to color big squares red,  
and small squares blue...



...while all circles are purple.



# NESTED CONDITIONALS

## Splitting results for a single condition

```
if ( *it's a square* ) {
```

```
    if ( *it's big* ) {
```

```
        *make it red!*
```

```
    } else {
```

```
        *it must be a small square, so make it blue!*
```

```
    }
```

```
} else {
```

```
    *since its not a square, it must be a circle, so make it purple!*
```

```
}
```

This Else ONLY reacts to a failure of the most recently encountered If statement

This Else ONLY triggers if the very first If does not.

# NESTED CONDITIONALS

## Splitting results for a single condition

```
if (*there are ANY running trains*) {  
    if (*the amount of running trains equals the amount of total trains*) {  
        *print out to passengers that all trains are running!*  
    } else {  
        *just execute our existing loop code covering the status of trains*  
    }  
} else {  
    *there must be no running trains, so print that out!*  
}
```



# NESTED CONDITIONALS

## Splitting results for a single condition

```
if (trainsOperational > 0) {  
    if (trainsOperational == totalTrains) {  
        console.log("All trains are running at the JavaScript Express!");  
    } else {  
        for (var trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
            ...  
        }  
    }  
} else {  
    console.log("No trains are operational today. Bummer!");  
}
```

... ← Our already existing status conditionals from trains.js go here

# UPDATING TRAINS.JS WITH NEW CONDITIONS

Now passengers will know if all trains are running, or if none are.

```
var totalTrains = 12;
var trainsOperational = 8;
if ( trainsOperational > 0 ) {
    if (trainsOperational == totalTrains) {
        console.log("All trains are running at the JavaScript Express!");
    } else {
        for (var trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
            ...
        }
    }
} else {
    console.log("No trains are operational today. Bummer!");
}
```

← Every possible message is still controlled by these two values!

**trains.js**





# ALL TRAINS, OR NONE

Let's look at how we'd get these situations...

```
var totalTrains = 12;  
var trainsOperational = 12;
```

trainsOperational > 0



trainsOperational == totalTrains



All trains are running at the JavaScript Express!



# ALL TRAINS, OR NONE

Let's look at how we'd get these situations...

```
var totalTrains = 12;  
var trainsOperational = 0;
```



```
trainsOperational > 0
```



No trains are operational today. Bummer!

# ADDING TO OUR LIST OF SPECIAL TRAINS

Another train that will start running at noon on its non-operational days

1



2



3



4



5



6



7



8



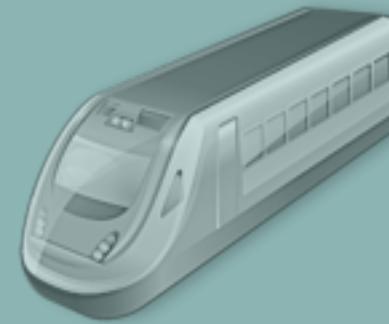
9



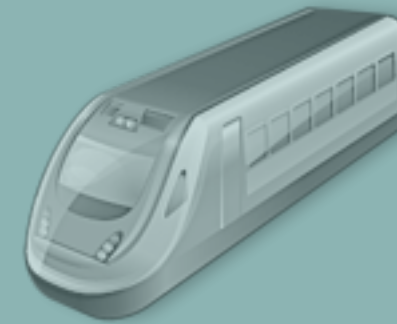
10



11



12



# ADDING TO OUR LIST OF SPECIAL TRAINS

Another train that will start running at noon on its non-operational days

1



2



3



4



5



6



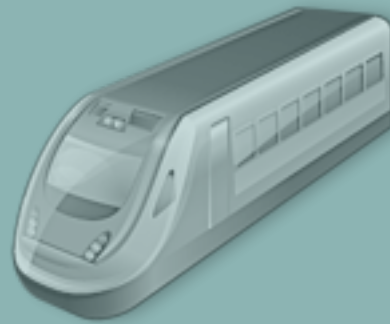
7



8



9



10



11



# ADDING TO OUR LIST OF SPECIAL TRAINS

Another train that will start running at noon on its non-operational days

1



2



3



4



5



6



7



8



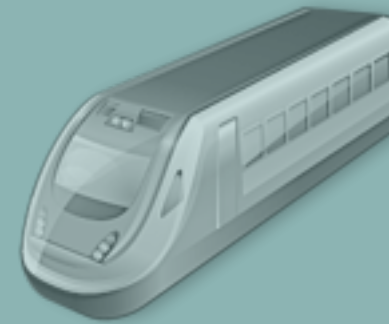
9



10



11



12



# ADDING A SECOND CONDITION

## Printing based on multiple conditions

trains.js

```
...  
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
    if (trainNumber <= operationalTrains) {  
        console.log("Train #" + trainNumber + " is running.");  
    } else if (trainNumber == 10 *we want something else here*) {  
        console.log("Train #10 begins running at noon.");  
    } else {  
        console.log("Train #" + trainNumber + " is not operational.");  
    }  
}
```



# COMPLEX CONDITIONALS

**&&** Binary 'And' returns true if BOTH values are true

**||** Binary 'Or' returns true if EITHER value is true

```
> true && false
```

→ false

```
> true && true
```

→ true

```
> false && false
```

→ false

```
> false || true
```

→ true

```
> false || false
```

→ false

```
> true || true
```

→ true



# COMPLEX CONDITIONALS

**&&** Binary 'And' returns true if BOTH values are true

**||** Binary 'Or' returns true if EITHER value is true

```
> ( 11 >= 11 ) && ( -7 < 6 )
```

true      &&      true

→ true

```
> ( 2 >= 0 ) && ( 9 < 4 )
```

true      &&      false

→ false

```
> ( 5 < 7 ) || ( 8 > 10 )
```

true      ||      false

→ true

```
> ( 3 > 8 ) || ( 7 < 3 )
```

false      ||      false

→ false





# ADDING A SECOND CONDITION

## Printing based on multiple conditions

trains.js

```
...  
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
    if (trainNumber <= operationalTrains) {  
        console.log("Train #" + trainNumber + " is running.");  
    } else if (trainNumber == 10 *we want something else here*) {  
        console.log("Train #10 begins running at noon.");  
    } else {  
        console.log("Train #" + trainNumber + " is not operational.");  
    }  
}
```



# ADDING A SECOND CONDITION

## Printing based on multiple conditions

trains.js
















```
...  
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
    if (trainNumber <= operationalTrains) {  
        console.log("Train #" + trainNumber + " is running.");  
    } else if (trainNumber == 10 || trainNumber == 12) {  
        console.log("Train #" + trainNumber + " will begin running at noon.");  
    } else {  
        console.log("Train #" + trainNumber + " is not operational.");  
    }  
}
```

Now we use the trainNumber variable instead of hard-coding the values.



# OUR UPDATED STATUS LOOP

All trains that begin running at noon are now printed!

trainNumber	LOOP: trainNumber<=12?	Is trainNumber<=8?	Is trainNumber ==10?	OUTPUT
 1	TRUE	YES -> IF	IGNORE	Train #1 is running.
 2	TRUE	YES -> IF		Train #2 is running.
 3	TRUE	YES -> IF		Train #3 is running.
 4	TRUE	YES -> IF		Train #4 is running.
 5	TRUE	YES -> IF		Train #5 is running.
 6	TRUE	YES -> IF		Train #6 is running.
 7	TRUE	YES -> IF		Train #7 is running.
 8	TRUE	YES -> IF		Train #8 is running.
 9	TRUE	NO	NO -> ELSE	Train #9 is not operational.
 10	TRUE	NO	YES -> ELSE-IF	Train #10 begins running at noon.
 11	TRUE	NO	NO -> ELSE	Train #11 is not operational.
 12	TRUE	NO	YES -> ELSE-IF	Train #12 begins running at noon.
 13	FALSE	STOP THE LOOP!		

# ADDING TO OUR LIST OF SPECIAL TRAINS

Using && for unique conditions

1



2



3



4



5



6



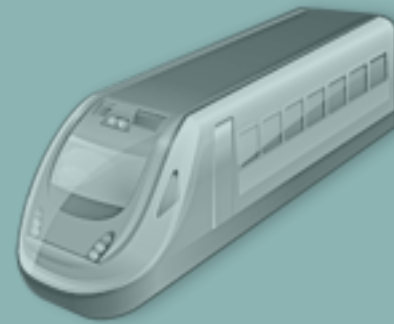
7



8



9



10



11



12



# ADDING TO OUR LIST OF SPECIAL TRAINS

Using && for unique conditions

1



2



4



5



6



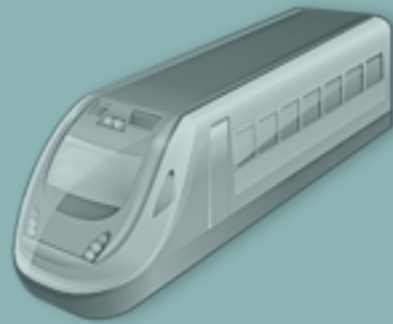
7



8



9



10



11



12



# ADDING TO OUR LIST OF SPECIAL TRAINS

Using && for unique conditions

1



2



3



4



5



6



7



8



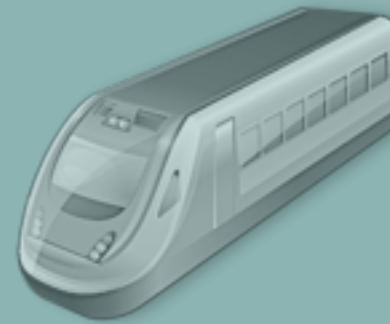
9



10



11



12



# INSERTING NEW CONDITIONS

We want to make sure Train 3 runs only on Sunday

```
var dayOfWeek = "Friday";
```

**trains.js**

```
...
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
    if (trainNumber <= operationalTrains) {
        console.log("Train #" + trainNumber + " is running.");
    } else if (trainNumber == 10 || trainNumber == 12 ) {
        console.log("Train # " + trainNumber + " will begin running at noon.");
    } else if ( *trainNumber is 3 AND its Sunday* ) {
        *print that train 3 is running*
    } else {
        console.log("Train #" + trainNumber + " is not operational.");
    }
}
```



# INSERTING NEW CONDITIONS

We want to make sure Train 3 runs only on Sunday

```
var dayOfWeek = "Friday";
```

```
...
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
    if (trainNumber <= operationalTrains) {
        console.log("Train #" + trainNumber + " is running.");
    } else if (trainNumber == 10 || trainNumber == 12 ) {
        console.log("Train # " + trainNumber + " will begin running at noon.");
    } else if ( trainNumber == 3 && dayOfWeek == "Sunday" ) {
        console.log("Train #3 is running.");
    } else {
        console.log("Train #" + trainNumber + " is not operational.");
    }
}
}
```

trains.js





# LETS RUN IT!



Train #1 is running.

Train #2 is running.

Train #3 is running.

Train #4 is running.

Train #5 is running.

Train #6 is running.

Train #7 is running.

Train #8 is running.

Train #9 is not operational.

Train #10 will begin running at noon.

Train #11 is not operational.

Train #12 will begin running at noon.

← Womp, womp...

# WHY DIDN'T WE GET THE RIGHT STATUS?

## Tracing our loop logic

```
var dayOfWeek = "Friday";  
var totalTrains = 12;  
var trainsOperational = 8;
```

trainsOperational > 0 ✓

trainsOperational == totalTrains ✗

trainNumber = 3

*In our status loop,  
trainNumber  
eventually becomes 3*

trainNumber <= operationalTrains ✓

→ Train #3 is running. ✗

*But it shouldn't be running,  
because it's Friday!*

# HOUSTON, WE HAVE A PROBLEM...

Our logic doesn't work! What do we need...?

```
var dayOfWeek = "Friday";
```

trains.js



```
...
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
  if (trainNumber <= operationalTrains) {
    console.log("Train #" + trainNumber + " is running.");
  } else if (trainNumber == 10 || trainNumber == 12) {
    console.log("Train # " + trainNumber + " will begin running at noon.");
  } else if (trainNumber == 3 && dayOfWeek == "Sunday") {
    console.log("Train #3 is running.");
  } else {
    console.log("Train #" + trainNumber + " is not operational.");
  }
}
```

If there are 3 or more operational trains, then the later Else-If will never be checked when the trainNumber is 3. Thus, our system says that Train 3 is running when it isn't!

# HOUSTON, WE HAVE A PROBLEM...

Our logic doesn't work! What do we need...?

```
var dayOfWeek = "Friday";
```

```
...
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {
  if (trainNumber <= operationalTrains *AND trainNumber is NOT 3*) {
    console.log("Train #" + trainNumber + " is running.");
  } else if (trainNumber == 10 || trainNumber == 12 ) {
    console.log("Train # " + trainNumber + " will begin running at noon.");
  } else if ( trainNumber == 3 && dayOfWeek == "Sunday" ) {
    console.log("Train #3 is running.");
  } else {
    console.log("Train #" + trainNumber + " is not operational.");
  }
}
}
```

trains.js



# HOUSTON, WE HAVE A PROBLEM...

Our logic doesn't work! What do we need...?

```
var dayOfWeek = "Friday";
```

trains.js



```
...  
for (trainNumber = 1; trainNumber <= totalTrains; trainNumber++) {  
    if (trainNumber <= operationalTrains && trainNumber !== 3 ) {  
        console.log("Train #" + trainNumber + " is running.");  
    } else if (trainNumber == 10 || trainNumber == 12 ) {  
        console.log("Train # " + trainNumber + " will begin running at noon.");  
    } else if ( trainNumber == 3 && dayOfWeek == "Sunday" ) {  
        console.log("Train #3 is running.");  
    } else {  
        console.log("Train #" + trainNumber + " is not operational.");  
    }  
}
```

If we have made sure that trainNumber is NOT 3 before printing out for a regular running train, this later Else-If will trigger correctly if both conditions are met!

# NOW WE GET CORRECT PRINTOUTS!

```
var dayOfWeek = "Friday";
```



Train #1 is running.  
Train #2 is running.  
Train #3 is not operational.  
Train #4 is running.  
Train #5 is running.  
Train #6 is running.  
Train #7 is running.  
Train #8 is running.  
Train #9 is not operational.  
Train #10 will begin running at noon.  
Train #11 is not operational.  
Train #12 will begin running at noon.

# NOW WE GET CORRECT PRINTOUTS!

```
var dayOfWeek = "Sunday";
```



Train #1 is running.  
Train #2 is running.  
Train #3 is running.  
Train #4 is running.  
Train #5 is running.  
Train #6 is running.  
Train #7 is running.  
Train #8 is running.  
Train #9 is not operational.  
Train #10 will begin running at noon.  
Train #11 is not operational.  
Train #12 will begin running at noon.

# TRACING OUR COMPLEX CONDITIONAL

How do we arrive at the different printouts for Train 3?

```
var dayOfWeek = "Friday";  
var totalTrains = 12;  
var trainsOperational = 8;
```

```
trainsOperational > 0
```

```
trainsOperational == totalTrains
```

```
trainNumber = 3
```

*In the loop, trainNumber eventually becomes 3*

```
trainNumber <= operationalTrains && trainNumber != 3
```

```
trainNumber == 10 || trainNumber == 12
```

```
trainNumber == 3 && dayOfWeek == "Sunday"
```

→ Train #3 is not operational.



# TRACING OUR COMPLEX CONDITIONAL

How do we arrive at the different printouts for Train 3?

```
var dayOfWeek = "Sunday";  
var totalTrains = 12;  
var trainsOperational = 8;
```

```
trainsOperational > 0
```

```
trainsOperational == totalTrains
```

```
trainNumber = 3
```

In the loop, trainNumber eventually becomes 3

```
trainNumber <= operationalTrains && trainNumber != 3
```

```
trainNumber == 10 || trainNumber == 12
```

```
trainNumber == 3 && dayOfWeek == "Sunday"
```

→ Train #3 is running.