Events and Feedback

No screens

Prof. Lydia Chilton

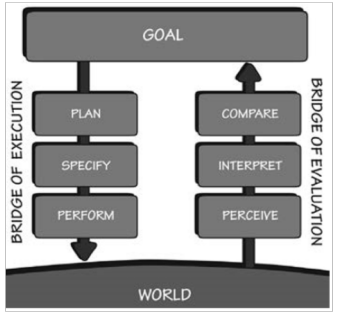
COMS 4170

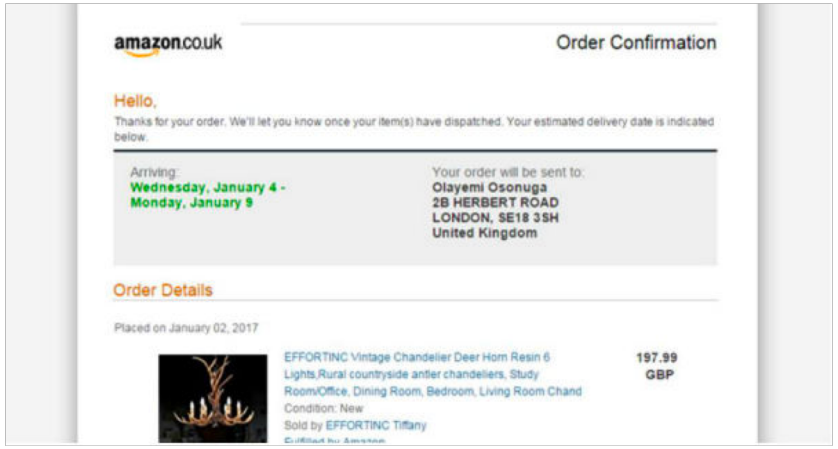
13 February 2019

Goal 1

Build websites that suit the needs and abilities of users

To accomplish a goal, users must
execute an operation and **evaluate** the result





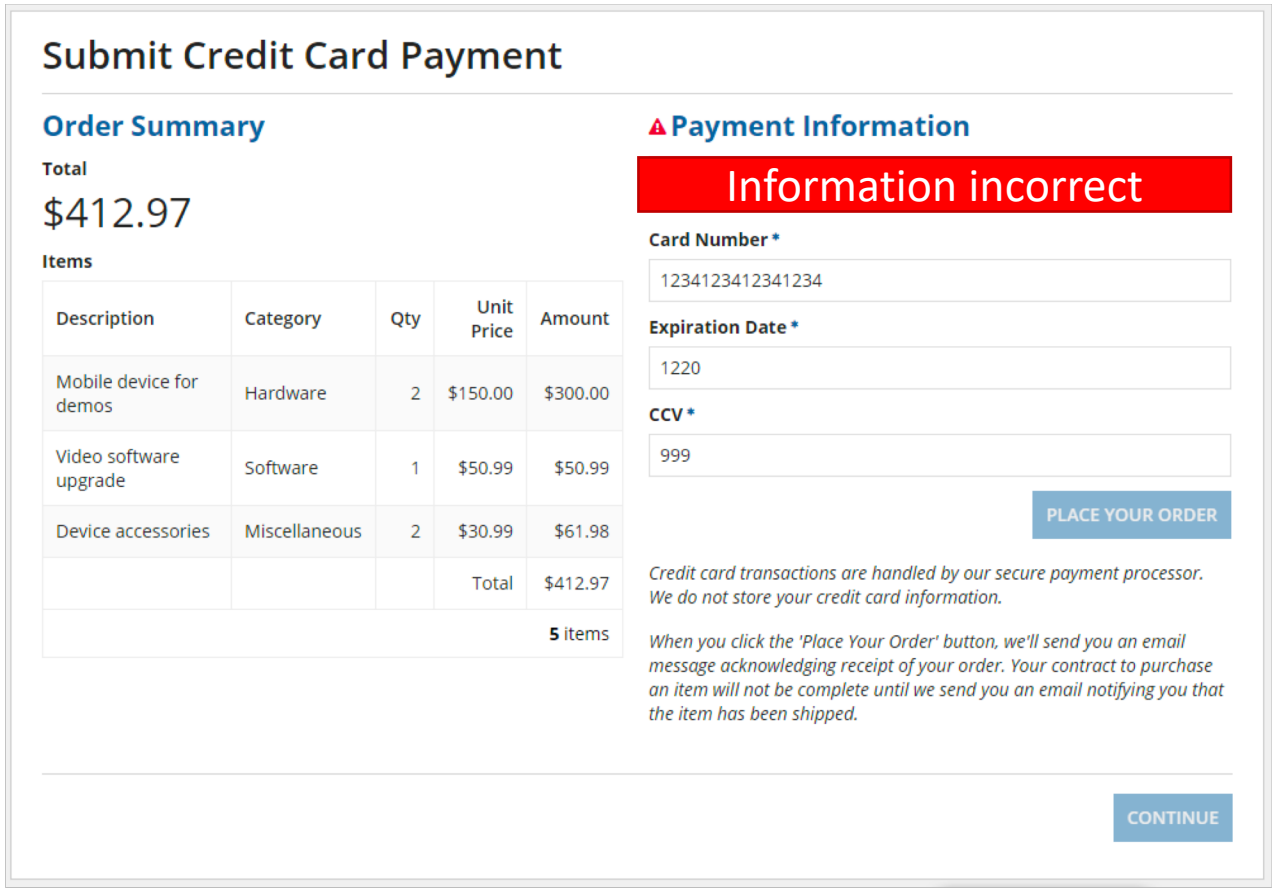
To help users **evaluate the result**,
designers must provide **feedback**.

What goes wrong when you provide no feedback?

Thought: Users are **confused** about whether their goal has been achieved.

Action: They continue to **expend energy** to **accomplish the goal**.

What goes wrong when you provide too little feedback?



Thought: Users know something has happened, but they **don't know what**.

Action: They continue to **expend energy** to **figure out what to do**.

else

What goes wrong when you provide too little feedback?

BEEP

Thought: Users know something has happened, but they **don't know what**.

Action: They continue to **expend energy** to **figure out** **if it's important**.

What goes wrong when you provide too much feedback?

I am now booking your flight

I am now using Google flight search

I am now typing JFK into the departure location

I am now typing LAX into the arrival location

I am now selecting February 26, 2018 from the departure date box

I am now confirming the date I just selected from the Departure date box

Thought: Users get **annoyed** that some of the information is useless

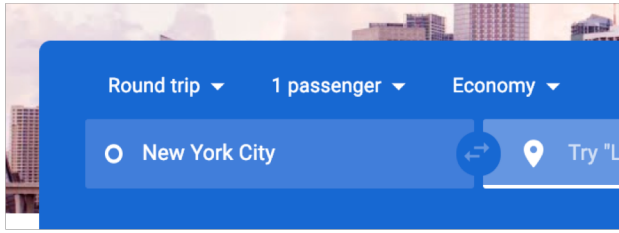
Action: They **ignore all the feedback**.

What goes wrong when feedback too late?

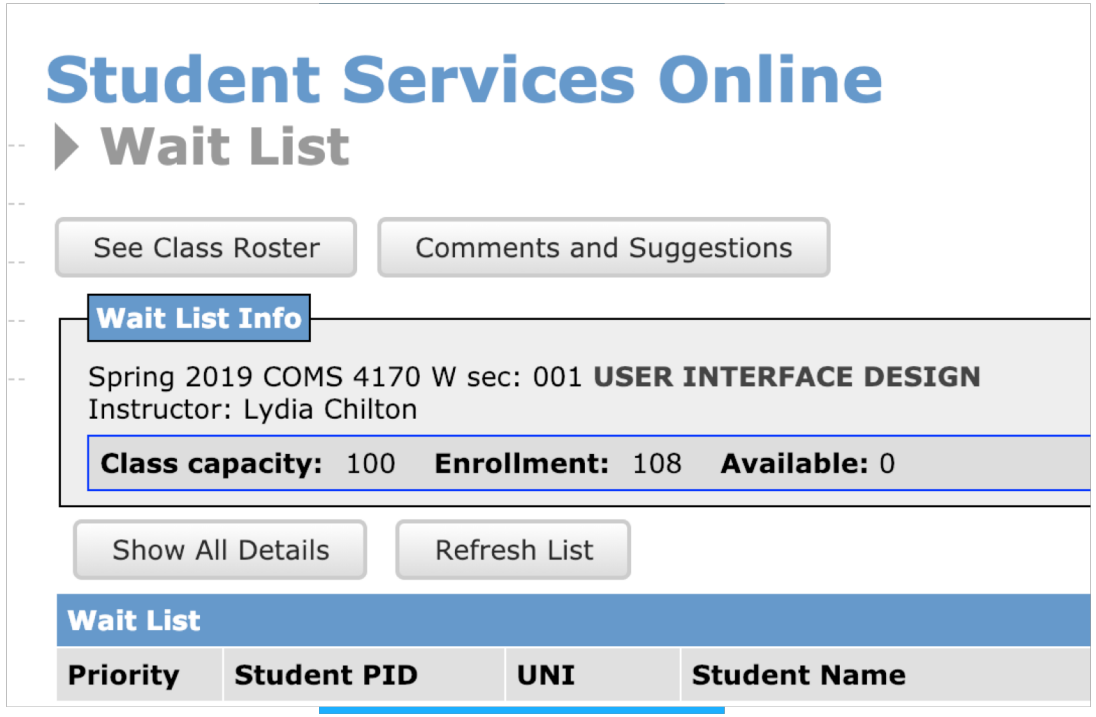
By the way, I booked that flight you asked for yesterday!

Thought: Users assume that **no feedback means no action**

Action: They **find another way** to reach their goal



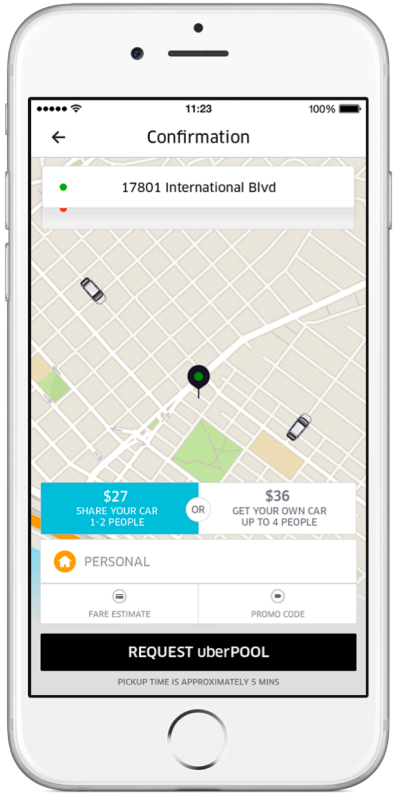
What goes wrong when feedback is not continuous?

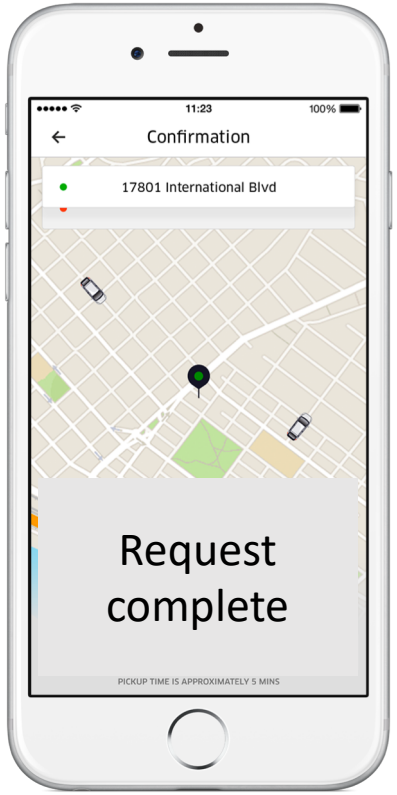


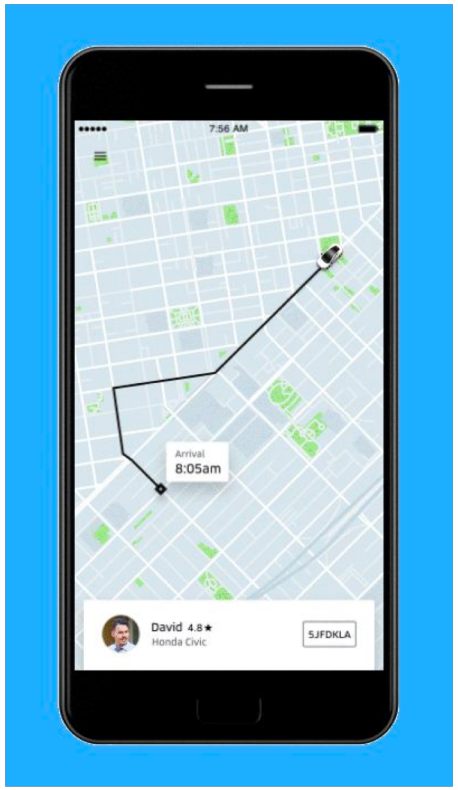
Thought: Users are **unsure** whether the system is doing it or not.

Action: Users have to **poll the system** for feedback frequently.

What goes wrong when feedback acknowledges the action but does not communicate the new state?







Thought: Users will think they are still in the old state.

Action: Users will continue to perform actions from the previous state

Design goals for feedback:

Communicate
full and continuous information about
the results of an action and
the current state of the system

to help people achieve their goal

Ways of perceiving feedback

How do we perceive this feedback?

BEEP

I’m sorry, Dave.
I’m afraid I can’t do that.

Examples of sound feedback?

How do we perceive this feedback?

Submit Credit Card Payment

Order Summary

Total

$412.97

Items

Description	Category	Qty	Unit Price	Amount
Mobile device for demos	Hardware	2	$150.00	$300.00
Video software upgrade	Software	1	$50.99	$50.99
Device accessories	Miscellaneous	2	$30.99	$61.98
			Total	$412.97

5 items

⚠ Payment Information

Information incorrect

Card Number *

1234123412341234

Expiration Date *

1220

CCV *

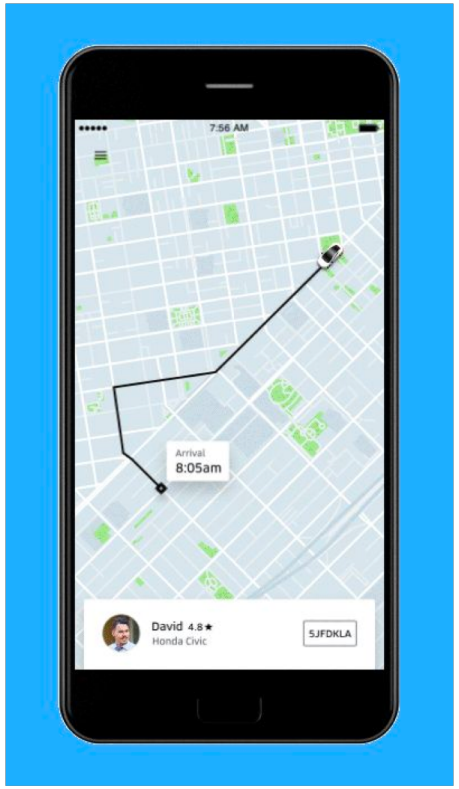
999

PLACE YOUR ORDER

Credit card transactions are handled by our secure payment processor. We do not store your credit card information.

When you click the 'Place Your Order' button, we'll send you an email message acknowledging receipt of your order. Your contract to purchase an item will not be complete until we send you an email notifying you that the item has been shipped.

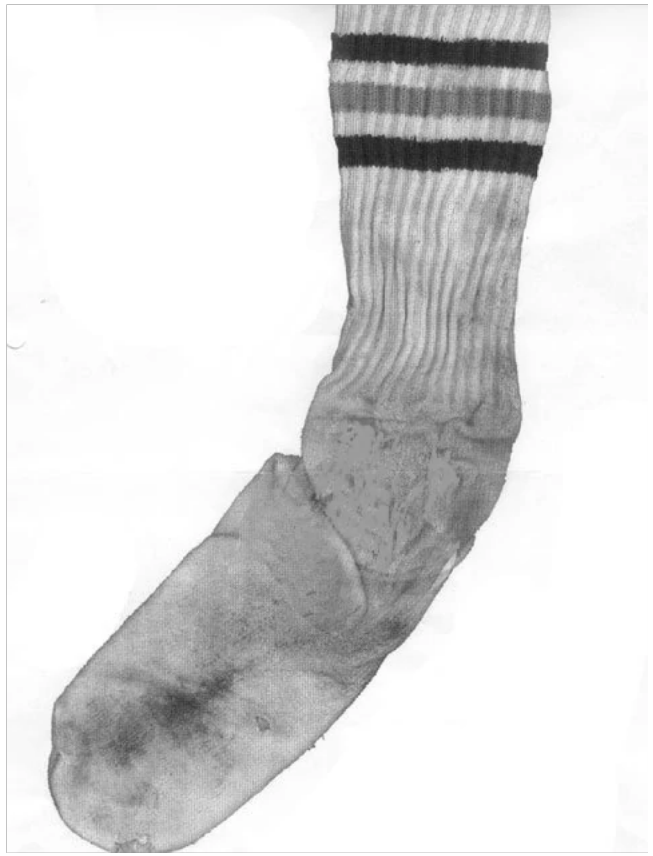
CONTINUE



Examples of visual feedback (non-digital)?

How do we perceive this feedback?

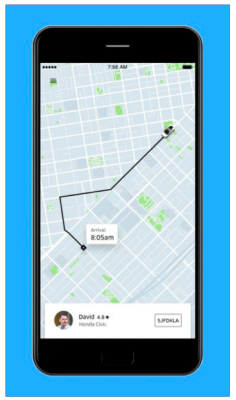
Examples of smell feedback?



How do we perceive this feedback?

Examples of haptic (touch) feedback?

The human nervous system is designed to perceive feedback in many forms.



Sight

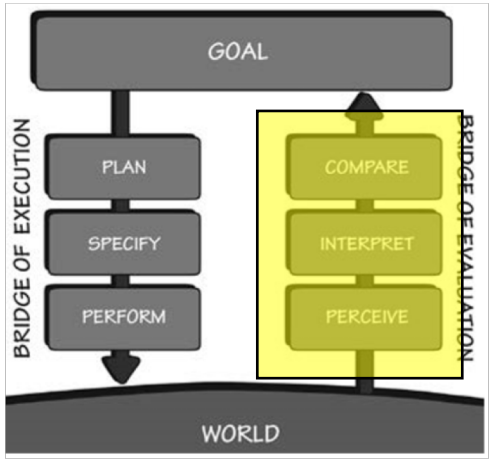
Sound

Smell

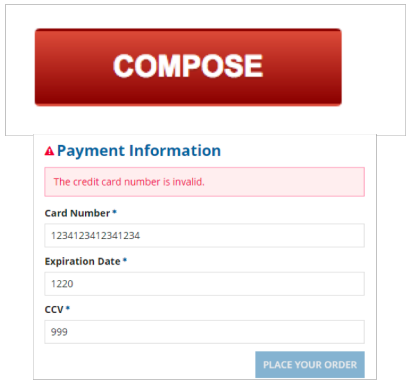
Touch

Physical Input Events and Feedback

Every time the user executes an action, the interface should provide feedback

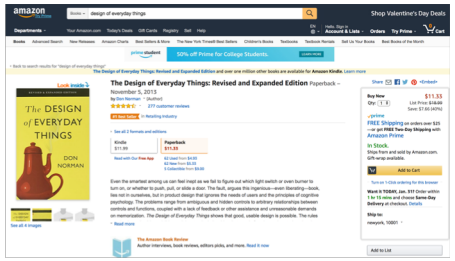


Low-level physical actions, like pressing a key



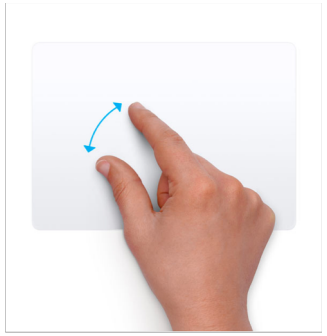
Low-level virtual actions, like clicking a button

Mid-level actions, like filling out a form



High-level actions, like buying a book

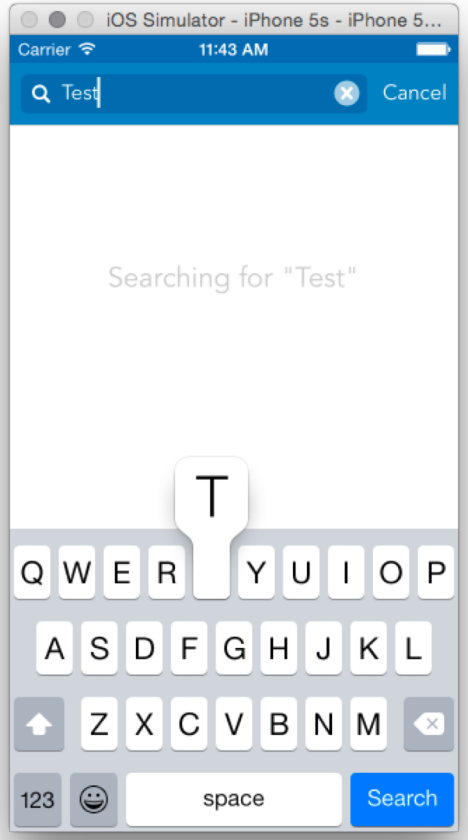
Low-level user actions are represented in the system as **events**.

Action	Event
	Keypress event
	Mousemove event Mousepress event
	Pinch gesture event

Types of hard keyboard **keypress** feedback?

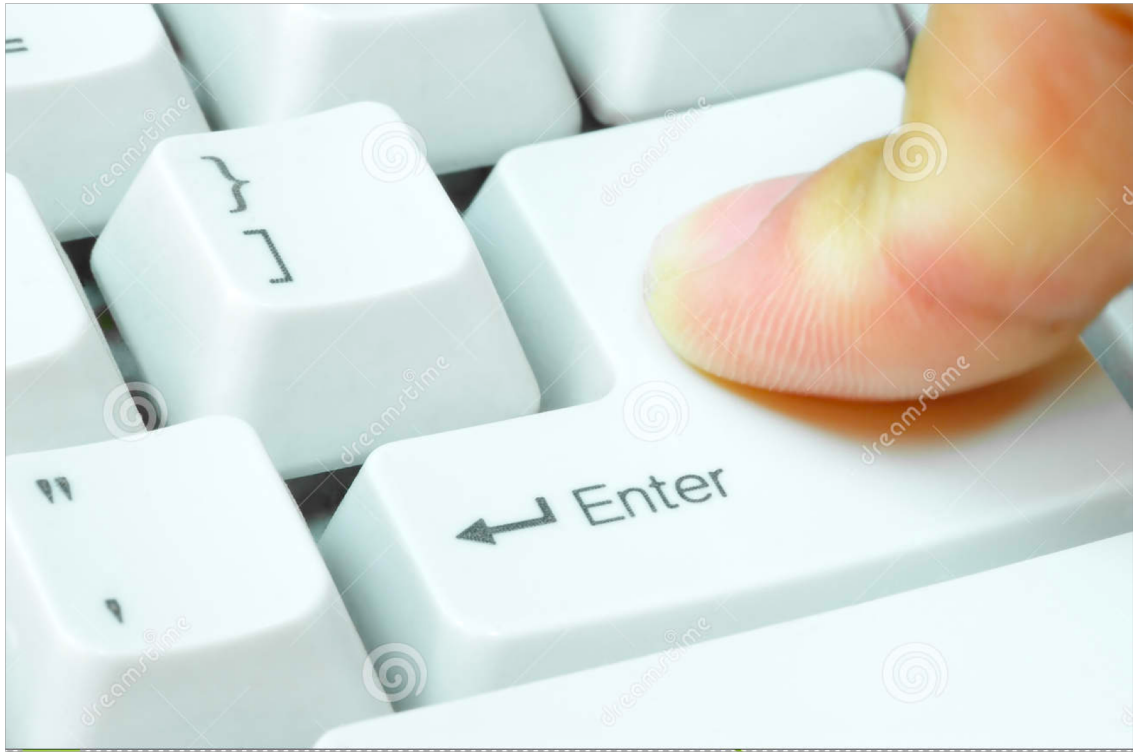
haptic (can feel the key), **sound**, and **visual** (screen action)

Types of soft keyboard **keypress** feedback?



~~haptic (can feel the key)~~, **sound (simulated)**, and **visual** (screen action)

Types of hard keyboard **key**down feedback?



haptic (can feel the spring pushing back),
sound (click), and
visual (can see it pushing down a tiny bit)

Types of hard keyboard **keyup** feedback?

haptic (can feel the spring pushing up),
sound (click), and
visual (can see it move a tiny bit)

Types of trackpad **Mousemove** feedback?

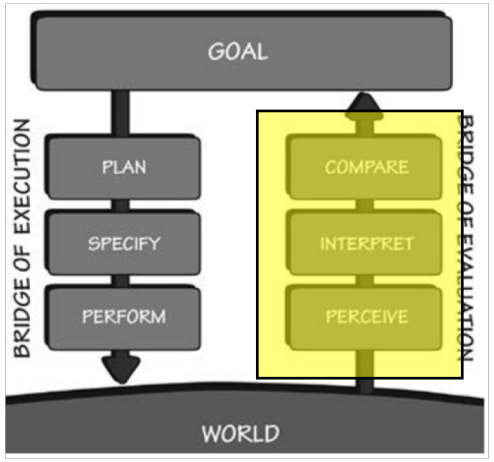
haptic (can feel the friction),
~~sound (no), and~~
visual (cursor movies)

Types of trackpad **Mousedown** feedback?

haptic (spring resistance),
sound (simulated click), and
visual (cursor movies)

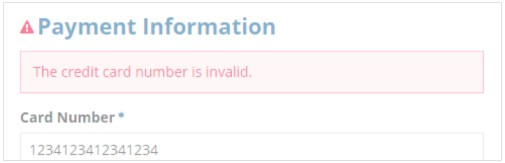
Low-level **Virtual** Feedback

Every time the user executes an action, the interface should provide feedback

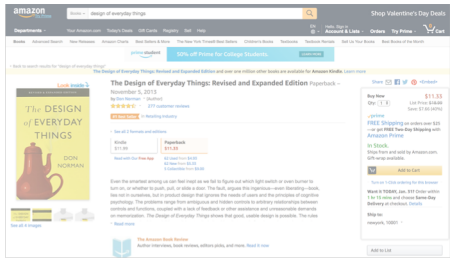


Low-level physical actions, like pressing a key

Low-level **virtual actions**, like clicking a button



Mid-level actions, like filling out a form

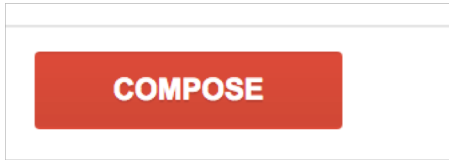


High-level actions, like buying a book

Subgoal: Press the compose button

What is the **first event** we respond to?

Normal state

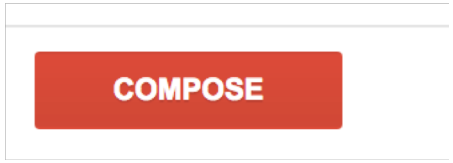



```
.T-I-KE {
  background-color: #d14836;
}
```

Subgoal: Press the compose button

What is the second **event** we respond to?

Normal state




```
.T-I-KE {
  background-color: #d14836;
}
```

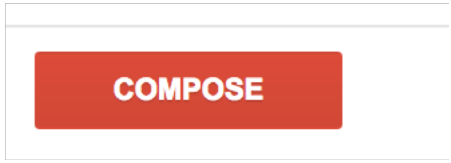
Mouseover feedback


```
.T-I-KE.T-I-JW {
  background-image: linear-gradient(to
    bottom, #dd4b39, #dd4b39);
}
```

Subgoal: Press the compose button

What is the third **event** we respond to?

Normal state




```
.T-I-KE {
  background-color: #d14836;
}
```

Mouseover feedback


```
.T-I-KE.T-I-JW {
  background-image: linear-gradient(to
    bottom, #dd4b39, #dd4b39);
}
```

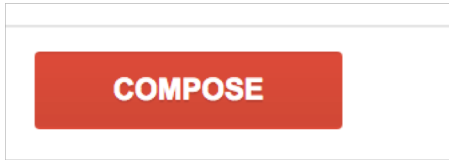
Mousedown feedback


```
background-image: linear-gradient(to
  bottom, #dd4b39, #400000);
```

Subgoal: Press the compose button

What is the final **event** we respond to?

Normal state




```
.T-I-KE {
  background-color: #d14836;
}
```

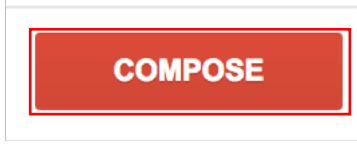
Mouseover feedback


```
.T-I-KE.T-I-JW {
  background-image: linear-gradient(to
    bottom, #dd4b39, #dd4b39);
}
```

Mousedown feedback


```
background-image: linear-gradient(to
  bottom, #dd4b39, #400000);
```

Mouseup feedback

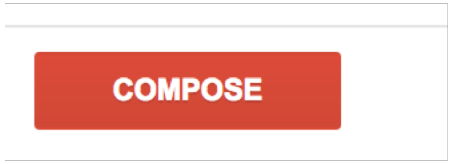



```
.T-I-KE {
  background-color: #d14836;
}
```

Implementing Low-level Feedback

How do you implement visual feedback?

Normal state




```
.T-I-KE {
  background-color: #d14836;
}
```

Mousedown


```
.T-I-KE.T-I-JW {
  background-image: linear-gradient(to
    bottom, #dd4b39, #400000);
}
```

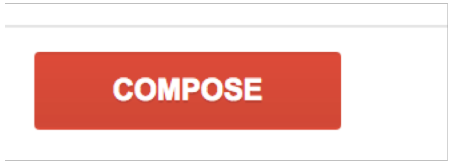
```
1  $(document).ready(function(){
2      $("#compose_button").mousedown(function(){
3
4          //?????????
5
6      })
7  })
```

1. Register an event handler on the object

2. Change the style

Can you change style like this?

Normal state




```
.T-I-KE {
   background-color: #d14836;
}
```

Mousedown

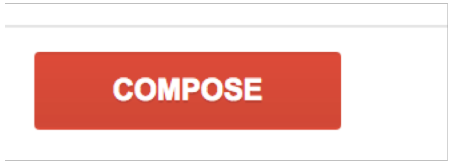

```
.T-I-KE.T-I-JW {
   background-image: linear-gradient(to
      bottom, #dd4b39, #400000);
}
```

```
1  $(document).ready(function(){
2      $("#compose_button").mousedown(function(){
3
4          $(this).css("background-image", "linear-gradient(to bottom, #dd4b38, #400000")
5
6      })
7  })
8
```

It will work, but it's **ugly**.

This is the better way to change style. Why?

Normal state




```
.T-I-KE {
  background-color: #d14836;
}
```

Mousedown

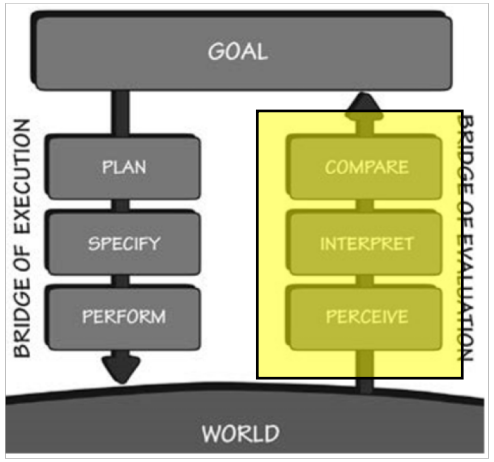

```
.T-I-KE.T-I-JW {
  background-image: linear-gradient(to
    bottom, #dd4b39, #400000);
}
```

```
1  $(document).ready(function(){
2      $("#compose_button").mousedown(function(){
3
4          //$(this).css("background-image", "linear-gradient(to bottom, #dd4b38, #400000")
5          $(this).addClass(    "T-I-JW"    )
6
7      })
8  })
```

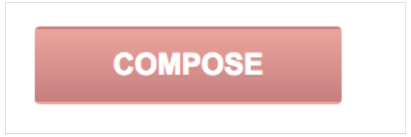
Classes abstract out designs and are easy to add / remove.

Mid-and High-level Action Feedback

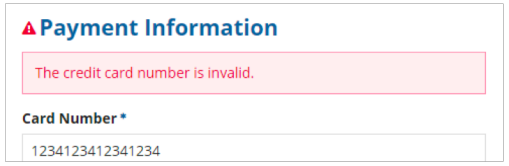
Every time the user executes an action, the interface should provide feedback



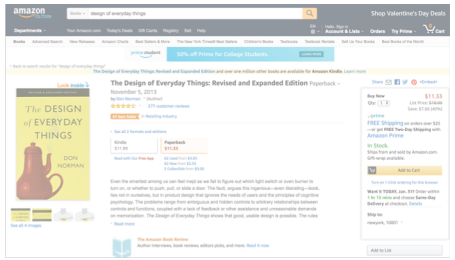
Low-level physical actions, like pressing a key



Low-level **virtual actions**, like clicking a button



Mid-level actions, like filling out a form



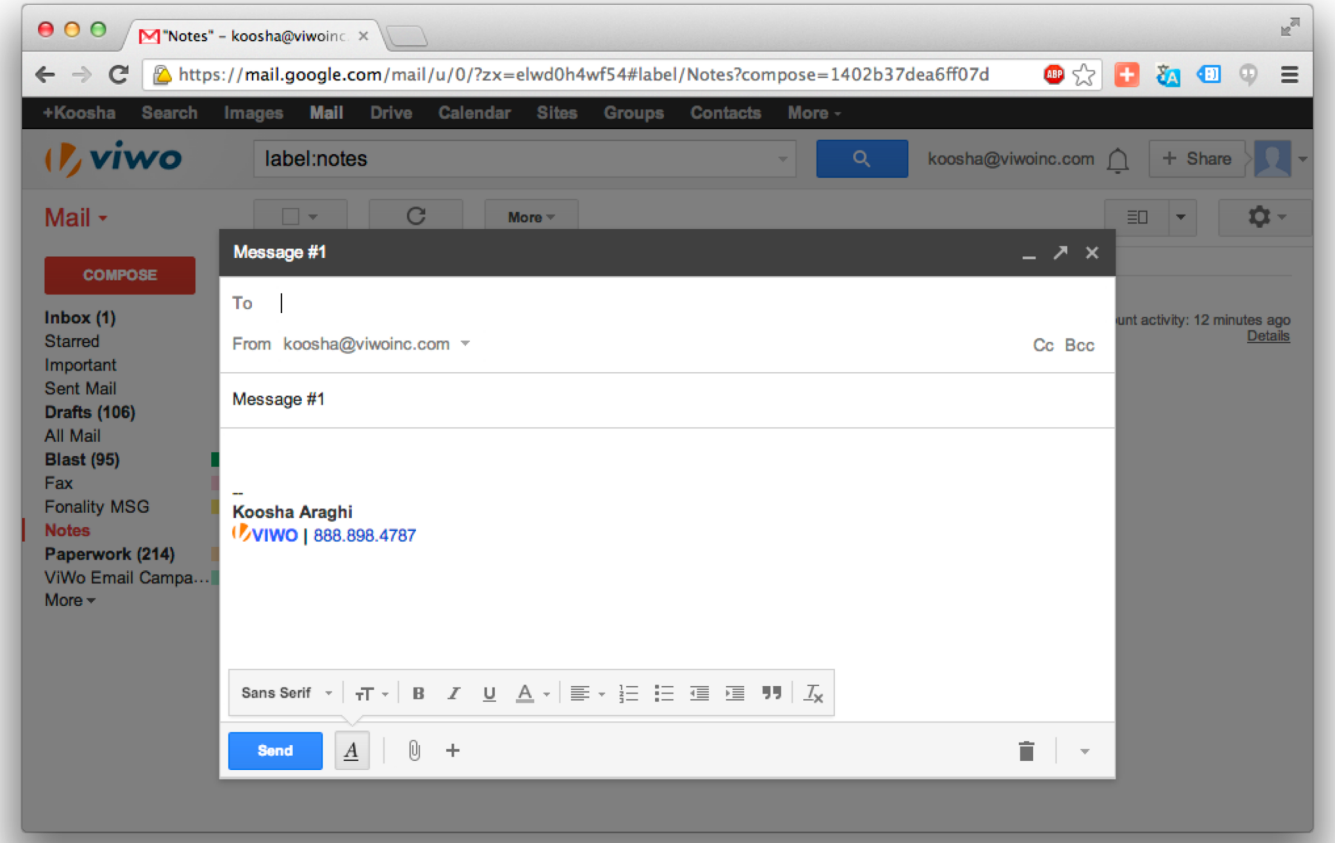
High-level actions, like buying a book

Feedback:

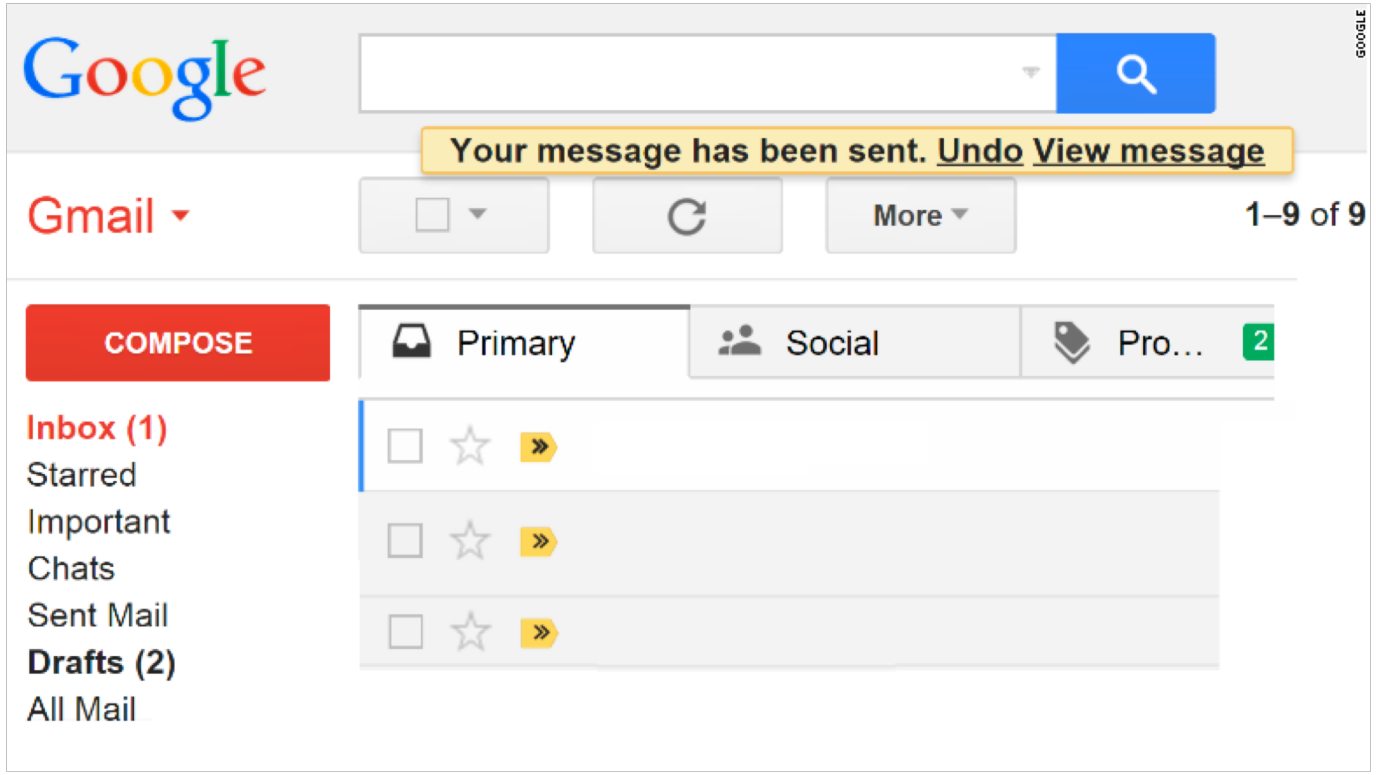
Communicate
full and continuous information about
the results of an action and
the current state of the system

to help people achieve their goal

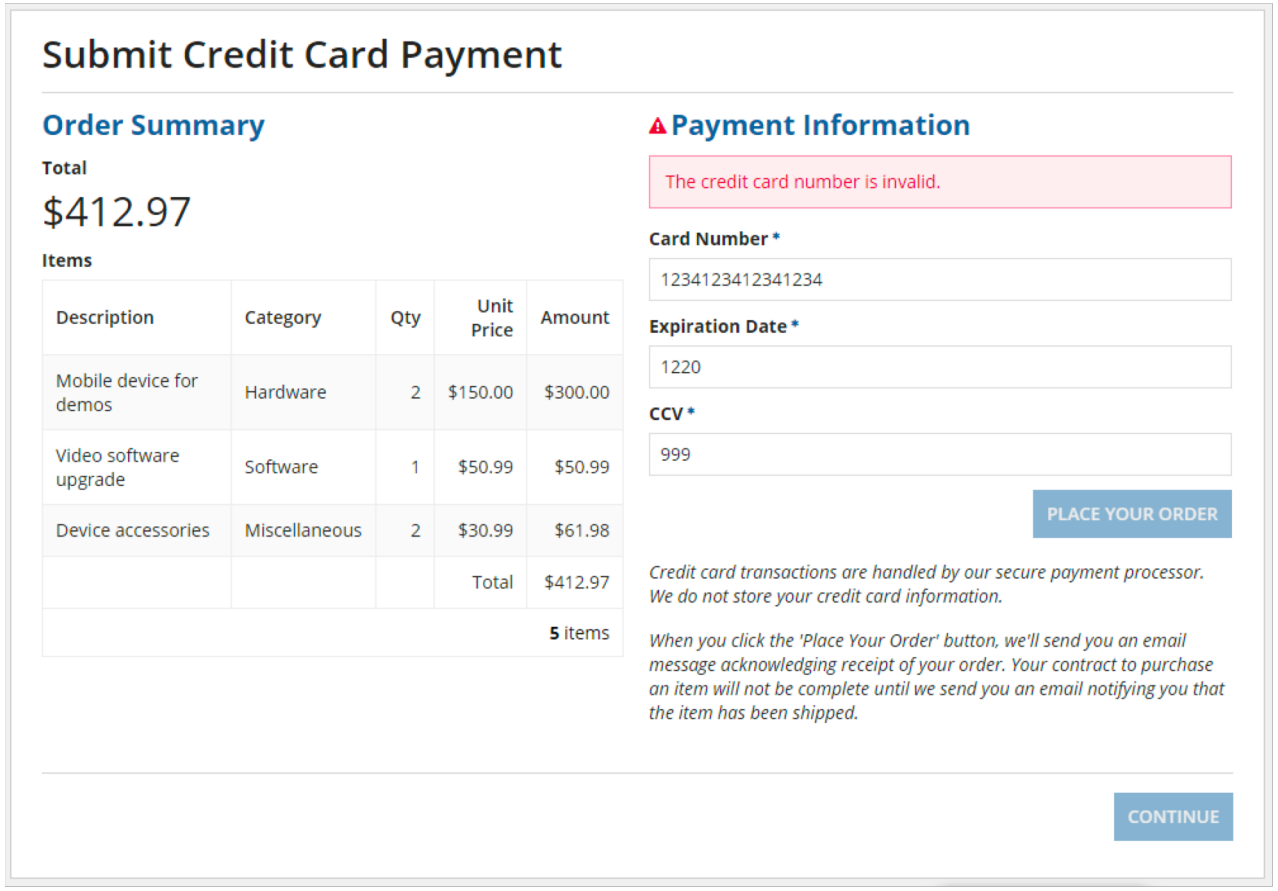
What is the new state?



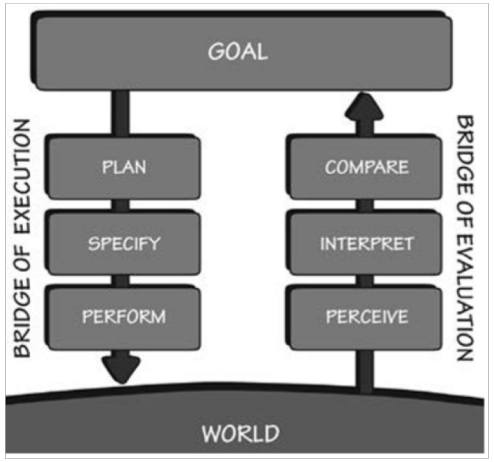
What is the new state?



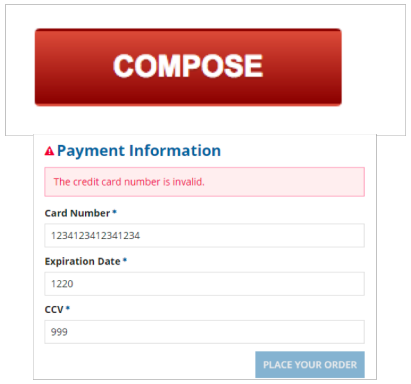
What is the new state?



Every time the user executes an action, the interface should provide feedback

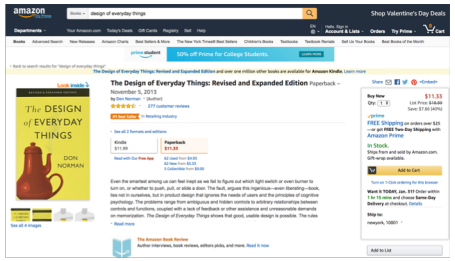


Low-level physical actions, like pressing a key



Low-level virtual actions, like clicking a button

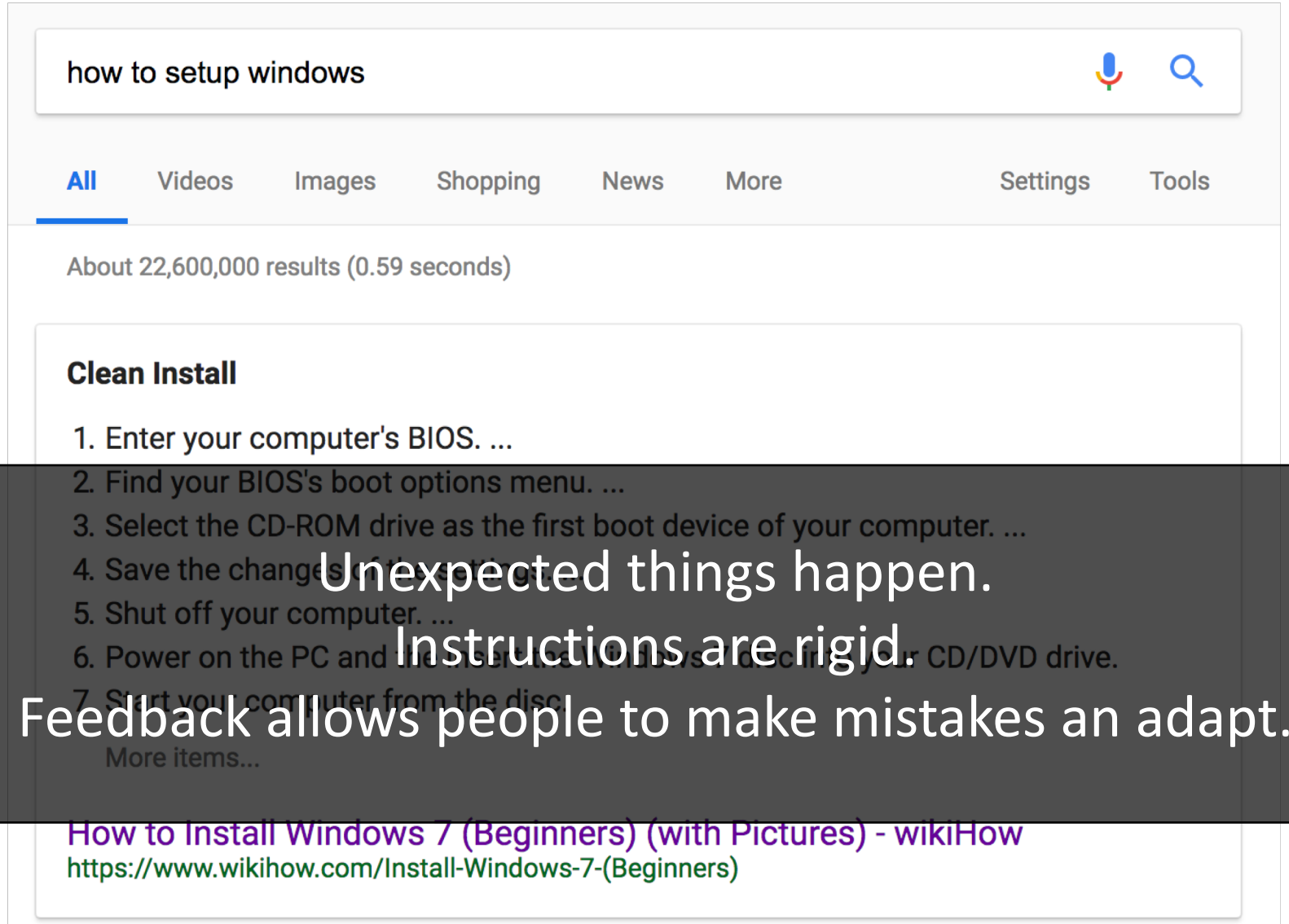
Mid-level actions, like filling out a form



High-level actions, like buying a book

Final Thought on Feedback

Following instructions sux. Why?



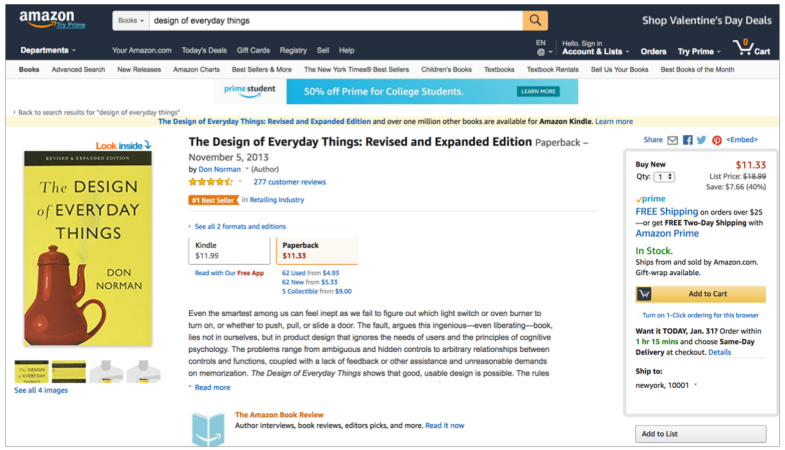
Unexpected things happen.

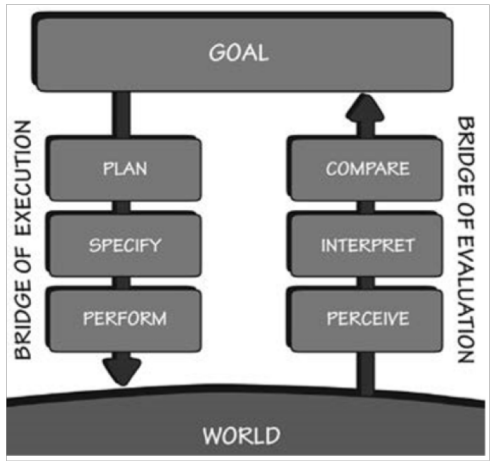
Instructions are rigid.

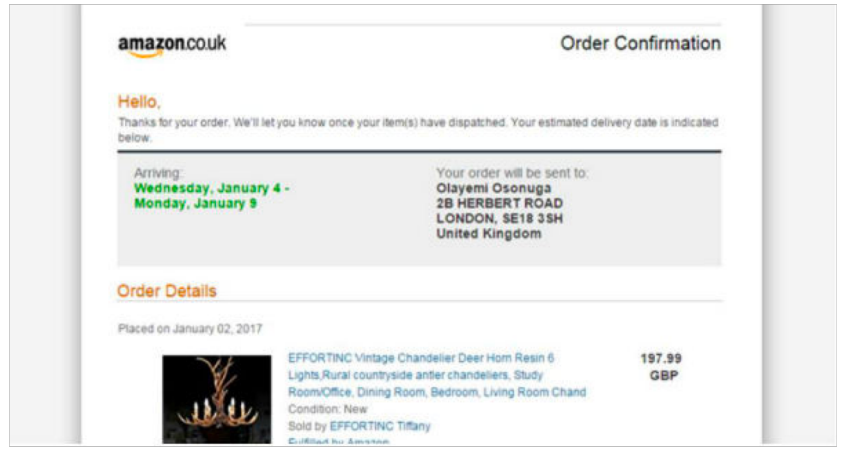
Feedback allows people to make mistakes an adapt.

Summary

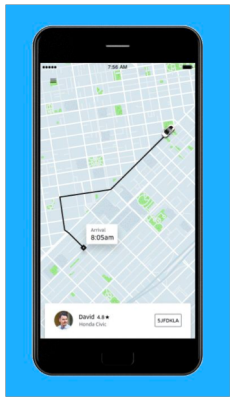
Feedback helps users evaluate the result of an action







The human nervous system is designed to perceive feedback in many forms.



Sight

Sound

Smell

Touch

Design feedback that:

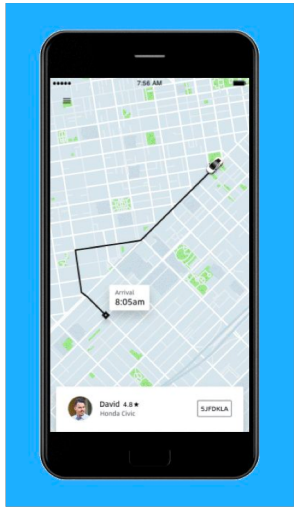
Communicates

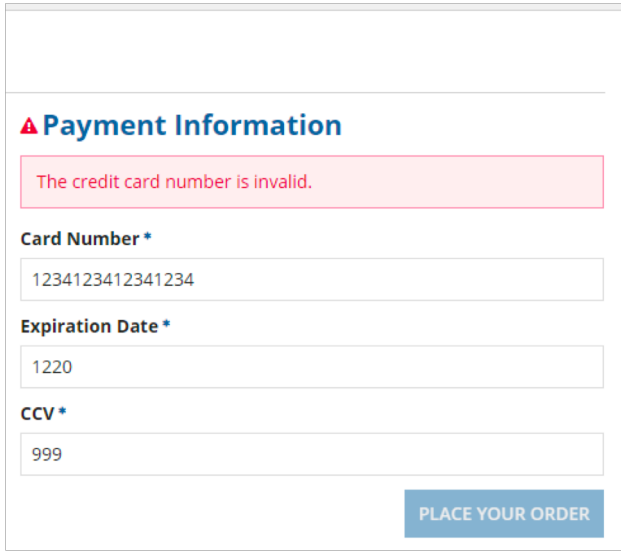
full and continuous information about the

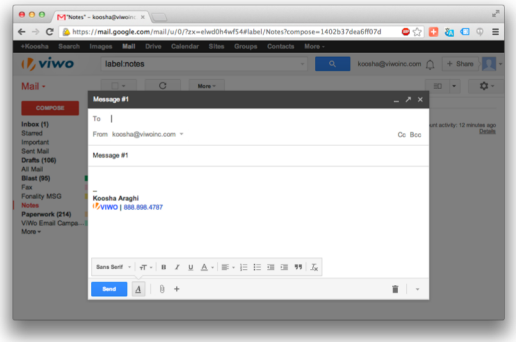
results of an action and the

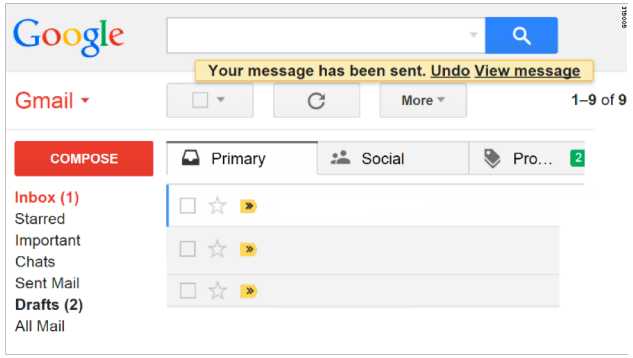
current state of the system

to help people achieve their goal

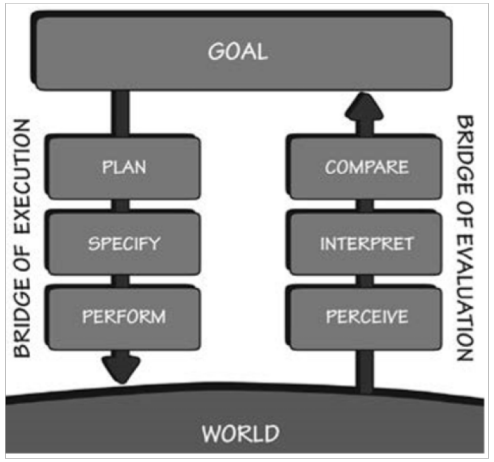




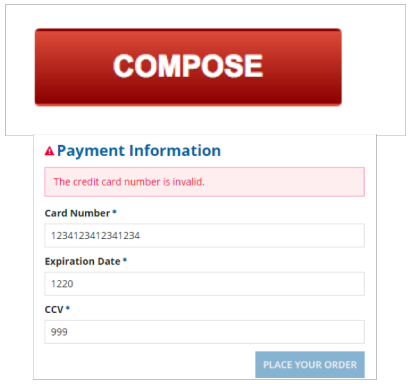




Every time the user executes an action, the interface should provide feedback

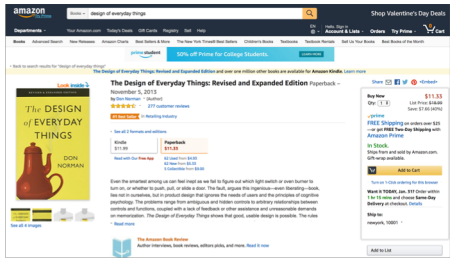


Low-level physical actions, like pressing a key



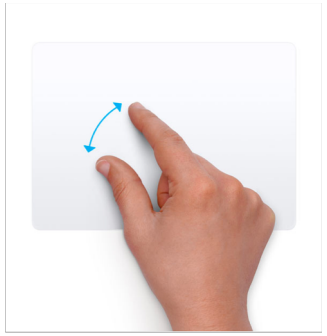
Low-level virtual actions, like clicking a button

Mid-level actions, like filling out a form

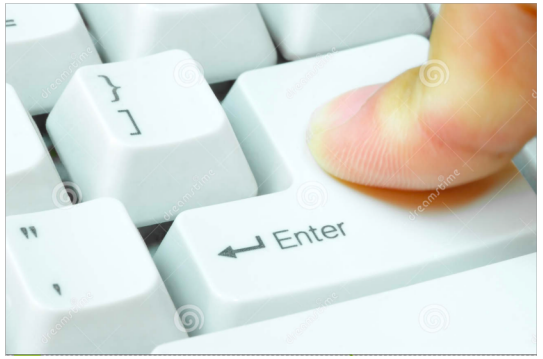


High-level actions, like buying a book

Low-level virtual actions are represented in the system as **events**.

Action	Event
	Keypress event
	Mousemove event Mousepress event
	Pinch gesture event

Even low-level events have **full and continuous feedback** about **actions and states**



Click!
Depress!

Normal state

COMPOSE

Mouseover

COMPOSE

Mousedown

COMPOSE

Mouseup

COMPOSE

Normal state

COMPOSE