Blocks are a Drag

Dragging Again

We have already learned how to the dragging gesture. We will now have a block that can be dragged around.

The Block component will keep track of its own dragging and movement.

We'll start with a simple state variable that keeps track of what is going on with the block.

Add this before the Block class.

```
enum DragState {
  NOT_DRAGGING,   // user not dragging anything
  DRAGGING_ME,    // user dragging this block
  DRAGGING_OTHER,// user dragging something else
}
```

As the player starts and stops dragging, each block will determine if it is being dragged.

Static Variables

What if we need to stop all blocks from being dragged?

- Maybe we need to drag something else first like our ball releaser
- We could have a variable in the Block component that we could turn dragging on and off
- We would have to go through every block and turn it on or off
- We could have a variable defined in the game itself that all blocks could check
- That means the game would have to know about blocks being dragged
- We want to keep dragging information all in one place
- How about defining a variable in Drag that all the objects could read together
- Static Variables

Define this variable before any instance variables in Block

```
// class variables apply to all blocks
static bool canDrag = false;  // set to true when we want blocks to be dragged
```

There are 2 types of variable inside a class definition

- Instance variable

  every object of that class has its own copy

- Static (or class) variable

  Only one copy exists, and all objects use the same one

- You can even set it if you don't have any objects

  Block.canDrag = false;

  All Block objects will now read false for this variable

- The canDrag static variable controls if any of the blocks respond to dragging

Handling the Drag

Add these instance variables that will keep track of dragging.

```
// handle block dragging
DragState dragState = DragState.NOT_DRAGGING;  // user not dragging
double dragX;   // x coordinate uf oser's finger
double dragY;   // y coordinate uf oser's finger
```

And what if we want just certain blocks to be draggable?

Like a special aiming block that the player can drag around

All others are stationary

Add this instance variable

```
final bool draggableBlock;  // if this block can be dragged
```

And add this variable to the named parameters of the constructor:

```
this.draggableBlock = false,
```

Review Dragging

- Our game object has a variable called isDragging
- It is set to true when user is dragging finger around
- Set to false when user lifts finger
- Game has two variables telling us where the finger Is dragging

  dragX, dragY
- Each component needs to keep track of when dragging starts. If it starts, and the user isn't dragging inside that component's rectangle, it ignores the whole drag
- If dragging starts and the component is being dragged, it keeps track of the finger until dragging stops

```
void update(double t) {
  // update position if being dragged
  if (canDrag && draggableBlock && game.isDragging) { // user currently dragging
    if (dragState == DragState.NOT_DRAGGING) {
      // nothing is being dragged yet, so it might be this block
      dragX = game.dragX; // record where dragging started
      dragY = game.dragY;
      if (position.contains(Offset(dragX,dragY))) {
        // we are being dragged
        dragState = DragState.DRAGGING_ME;
      } else {
        // something else being dragged
        dragState = DragState.DRAGGING_OTHER;
      }
    } else if (dragState == DragState.DRAGGING_ME) {
      // this block was already being dragged, so move it
      position = position.translate(game.dragX - dragX, game.dragY - dragY);
      dragX = game.dragX; // record last finger position
      dragY = game.dragY;
    }
  } else {
    // user not dragging
    dragState = DragState.NOT_DRAGGING;
  }
}
```

Update Explained

```
if (canDrag && draggableBlock && game.isDragging) { // user currently dragging

...

} else {
  // user not dragging
  dragState = DragState.NOT_DRAGGING;
}
```

We only do dragging if all blocks can be dragged, this block can be dragged, and the user is actually dragging their finger. Otherwise, we set the state to NOT_DRAGGING

NOT_DRAGGING will stop an existing drag if it is going on

Next, we have to see if dragging just started, if we are being dragged, or if someone else is being dragged

```
if (dragState == DragState.NOT_DRAGGING) {
```

This asks if we are not dragging right now. If we are not dragging this block, but the game is dragging, we have to figure out what is going on.

Either someone else is being dragged or we are about to start dragging this block

```
dragX = game.dragX; // record where dragging started
dragY = game.dragY;
if (position.contains(Offset(dragX,dragY))) {
  // we are being dragged
  dragState = DragState.DRAGGING_ME;
} else {
  // something else being dragged
  dragState = DragState.DRAGGING_OTHER;
}
```

We keep track of where the user's finger is

We then determine if we are being dragged or if someone else is being dragged.

We see if the finger is inside the block or not

The Actual Drag

```
} else if (dragState == DragState.DRAGGING_ME) {
  // this block was already being dragged, so move it
  position = position.translate(game.dragX - dragX, game.dragY - dragY);
  dragX = game.dragX; // record last finger position
  dragY = game.dragY;
}
```

This does the actual dragging of the block.

The variables dragX and dragY have the location of the finger on the previous update

The variables game.dragX and game.dragY have the location of the finger right now

You subtract the two to see how much the finger has moved from last update to this.

You move the block's position rectangle by the amount the finger moved

Then you update dragX and dragY to where the finger is right now so we can test the move next update

Try it Out

We will add 2 blocks to the game – one that can be dragged and one that cannot.

```
// make new block game components
var block = Block(this, position: Rect.fromLTWH(100, 200, 50, 50), draggableBlock:
true);
var block2 = Block(this, position: Rect.fromLTWH(200, 400, 50, 50), draggableBlock:
false);

// tell the game about the blocks
add(block);
add(block2);
Block.canDrag = true;  // ok to drag blocks
```

Try putting 2 blocks really close together – they will both drag together

The code is here

https://github.com/shawnlg/flutter_ball/tree/14_drag_block

Next time – bounce off of blocks