

平成17年!一日大学院



スクイークで コンピュータを 操ろう!!

知識科学教育研究センター
金井!秀明



授業の流れ



- コンピュータを操るには？
- スクイークって何？
- プロジェクト
 - 車を作ろう
 - 車を運転しよう
 - 利口な車を作ろう
- まとめ



コンピュータを操るには？

- ソフトウェア（ソフト）
 - 市販のソフト（ワープロ、ゲームなど）を使う
 - 自分で作ったソフトを使う。

「自分でソフトを作る」
＝「コンピュータプログラミング：
プログラミング」

3

プログラミングは？

- 「コンピュータをどのように動かしたいか」（命令）を書く。
- 命令は、プログラミング言語（BAISC, IC言語、Java言語...）で書きます。

4

例：プログラミング

The screenshot shows a Java code editor window titled "HelloUniverse.java" and a 3D rendering window titled "HelloUniverse". The code implements a 3D cube with alternating red and blue faces. The rendering window shows a perspective view of the cube against a black background.

```
TransformerGroup objTrans = new TransformerGroup();
objTrans.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
objRoot.addChild(objTrans);

// Create a simple Shape3D node; add it to the scene graph.
objTrans.addChild(new ColorCube(0.4));

// Create a new Behavior object that will perform the
// desired operation on the specified transform and add
// it into the scene graph.
Transformer3D yAxisis = new Transformer3D();
Alpha rotationAlpha = new Alpha(-1, 4000);

RotationInterpolator rotator =
    new RotationInterpolator(rotationAlpha, objTrans, yAxisis,
        new Point3D(0, 1, 0), (float) Math.PI*2.0f);
new BoundingSphere bounds(new Point3d(0,0,0,0), 100.0);
rotator.setSchedulingBounds(bounds);
objRoot.addChild(rotator);

// Have Java 3D perform optimizations on this scene graph.
objRoot.compile();

return objRoot;
}

public HelloUniverse() {
}

public void init() {
    setLayout(new BorderLayout());
    GraphicsConfiguration config =
        SimpleUniverse.getPreferredConfiguration();
    Canvas3D c = new Canvas3D(config);
    add("Center", c);

    // Create a simple scene and attach it to the virtual universe
    BranchGroup scene = createSceneGraph();
    u = new SimpleUniverse();
    u.addBranchGraph(scene);

    // This will move the ViewPlatform back a bit so the
    // objects in the scene can be viewed.
    u.getViewingPlatform().setNominalViewingTransform();
}

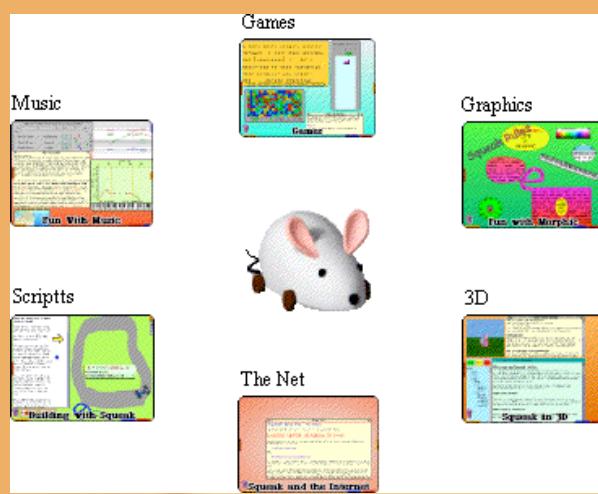
// Create a simple scene and attach it to the virtual universe
BranchGroup scene = createSceneGraph();
u = new SimpleUniverse();

// Add the scene to the ViewPlatform
u.addBranchGraph(scene);
}
```

5

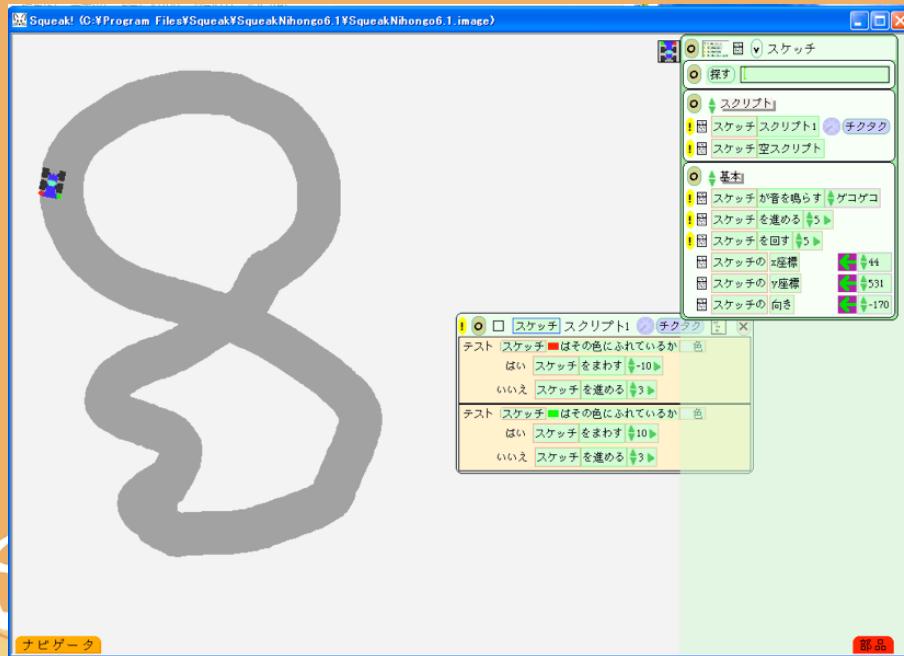
スクリークって何？

- プログラミング言語です。
- 例



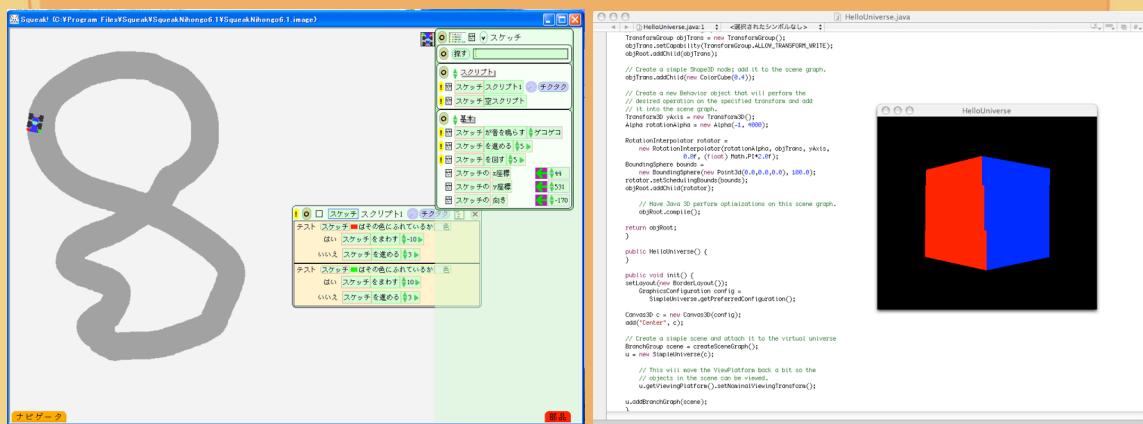
6

スクリプトの例



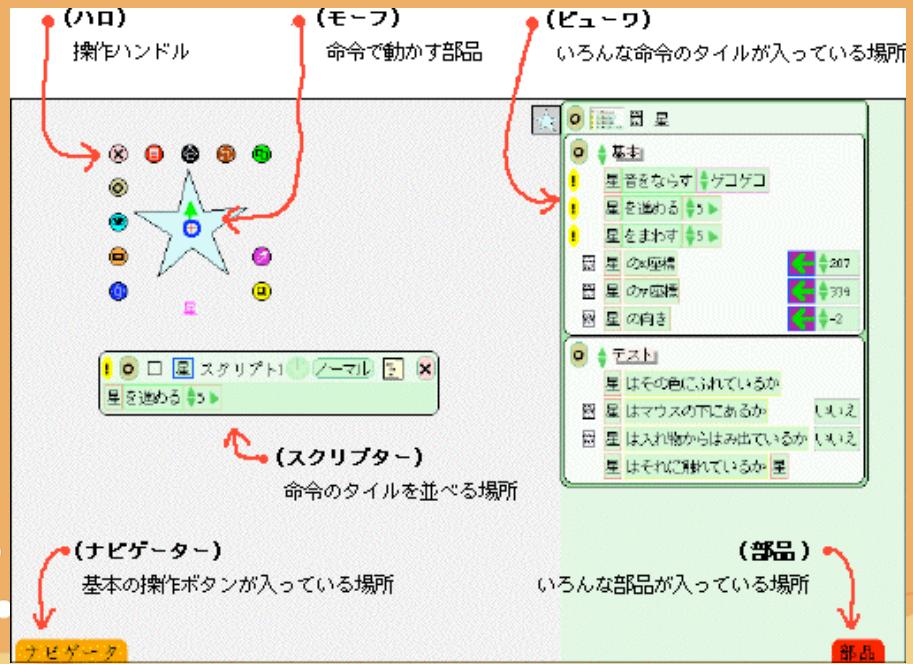
7

スクリプトと他のプログラム言語



8

スクイークの画面



9

今日の授業では、

- 「スクイーク」を使って、プログラミングを勉強します。
- ほとんど文字（命令）は書きません。
- 絵を描いて、その絵がどう動くかを指定するだけ。
- コンピュータの中で、車を動かすことができます。

10

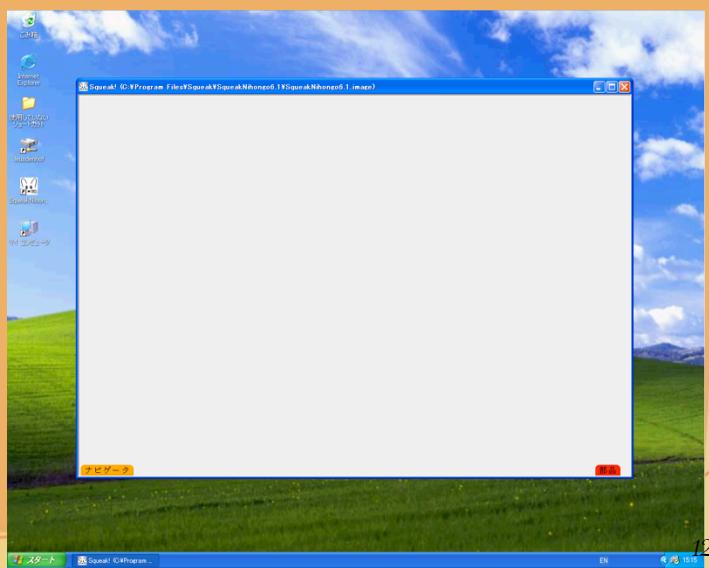
プロジェクト

- 車を作ろう
- 車を運転しよう
- 利口な車を作ろう

11

準備：スクイークの起動方法

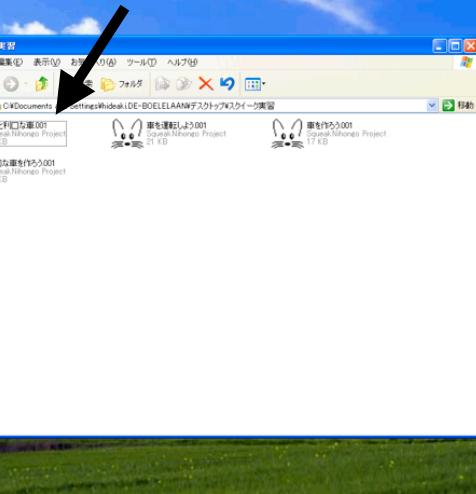
- Squeakのアイコンをクリックする。



12

実習例

このアイコンをクリックする



13

車を作ろう

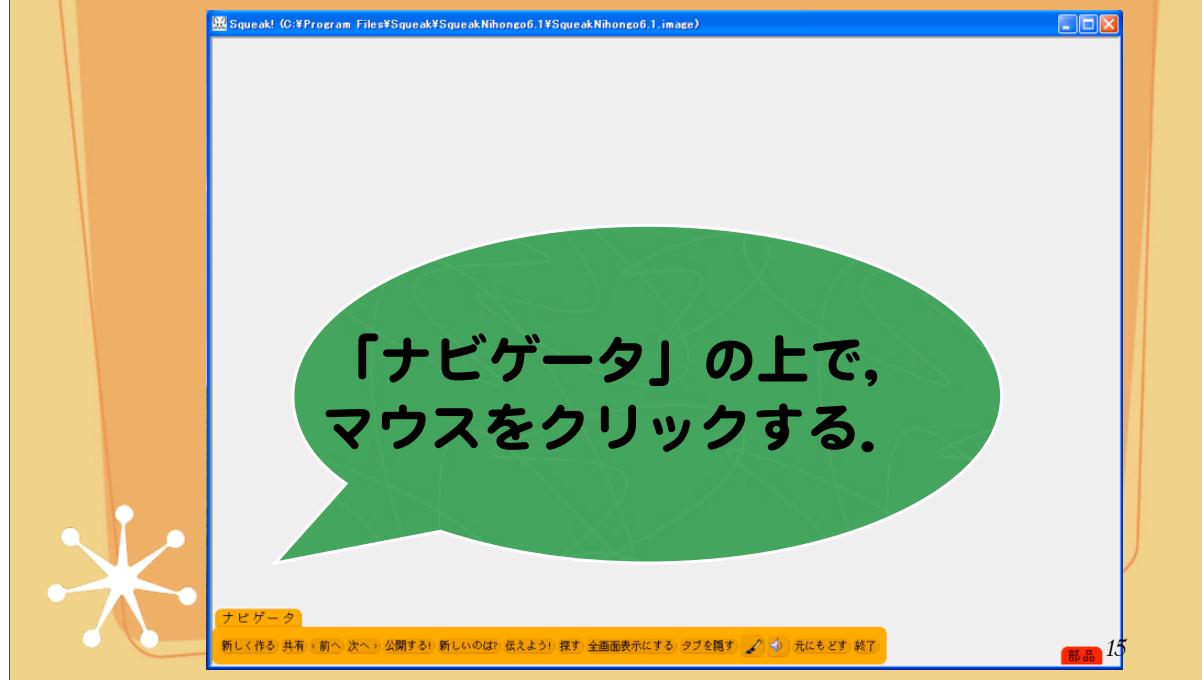


新しく作る 共有(前へ) 次へ) 公開する! 新しいのは? 伝えよう! 探す 全画面表示にする タブを隠す 元にもどす 終了

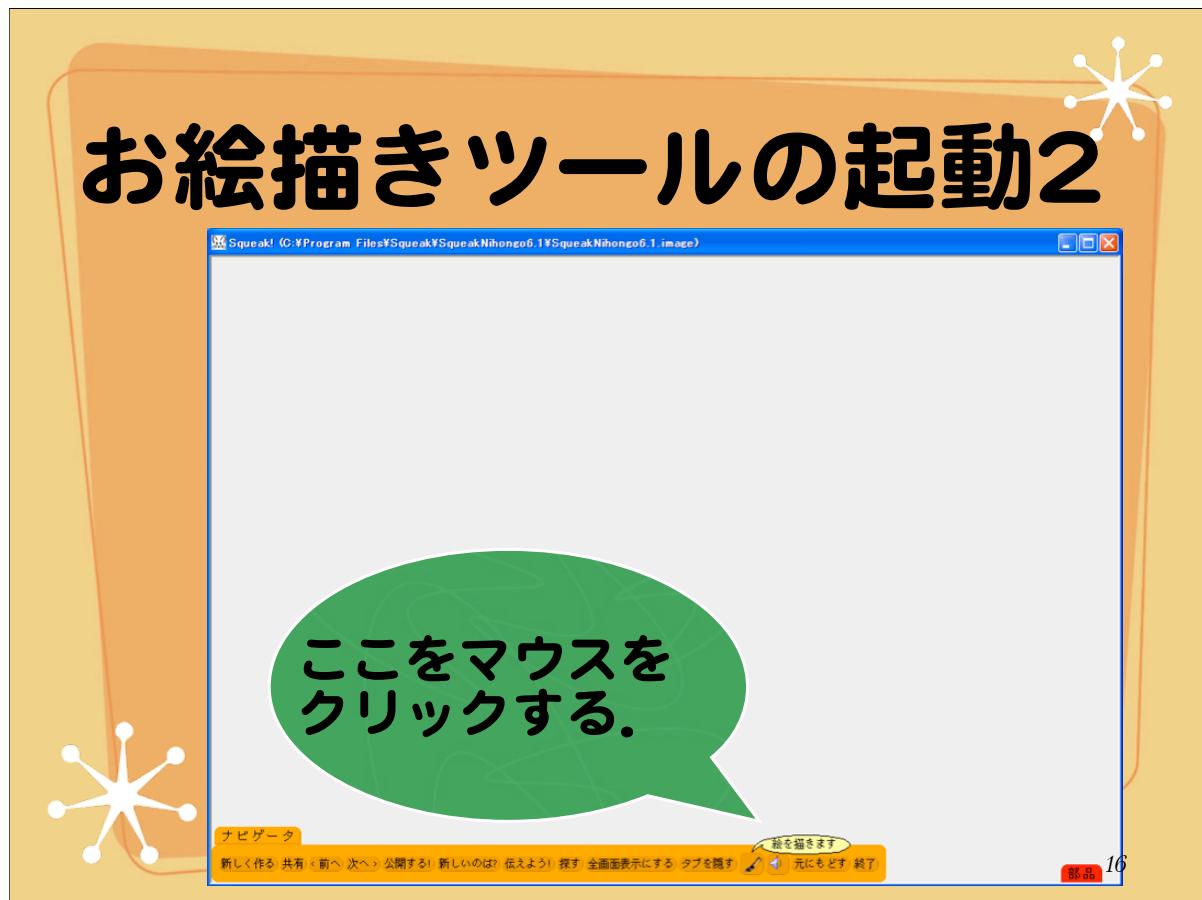
部品

14

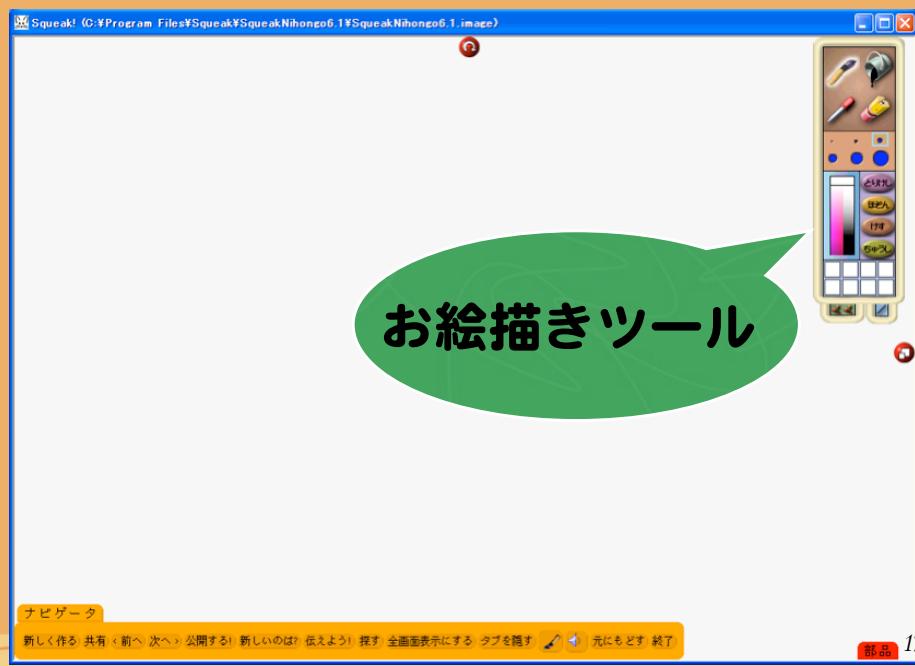
お絵描きツールの起動 1



お絵描きツールの起動 2



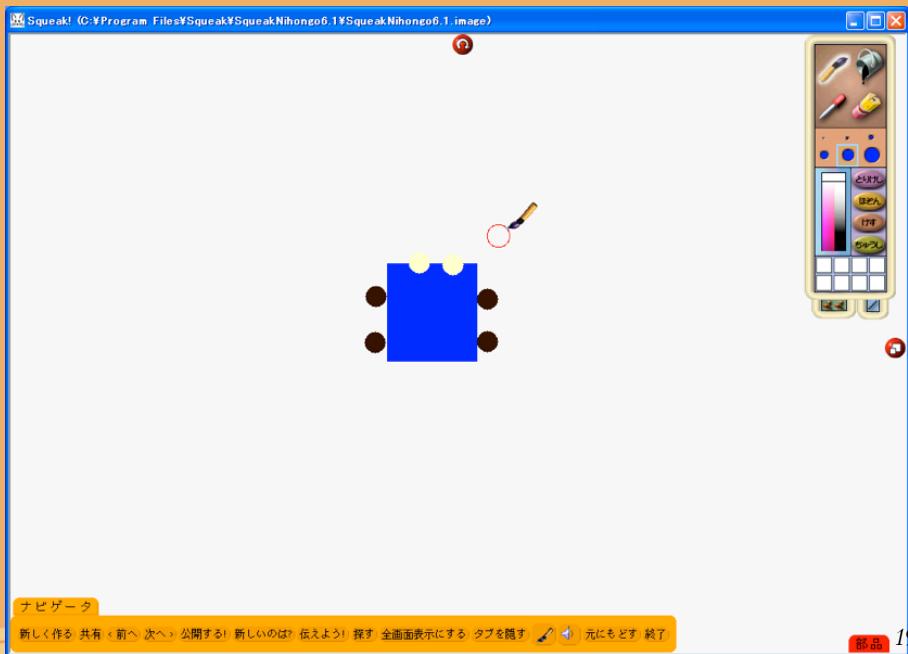
お絵描きツールの起動3



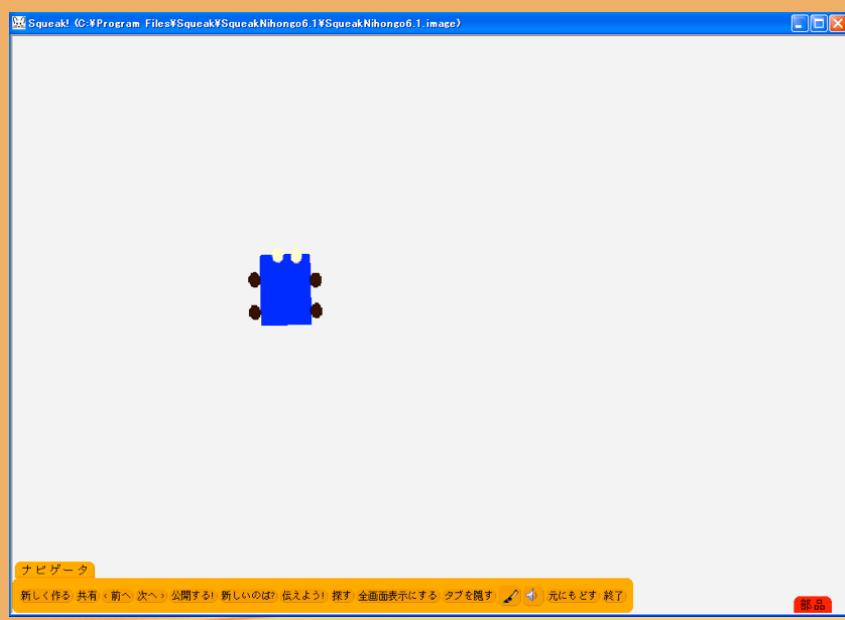
お絵描きツールの説明



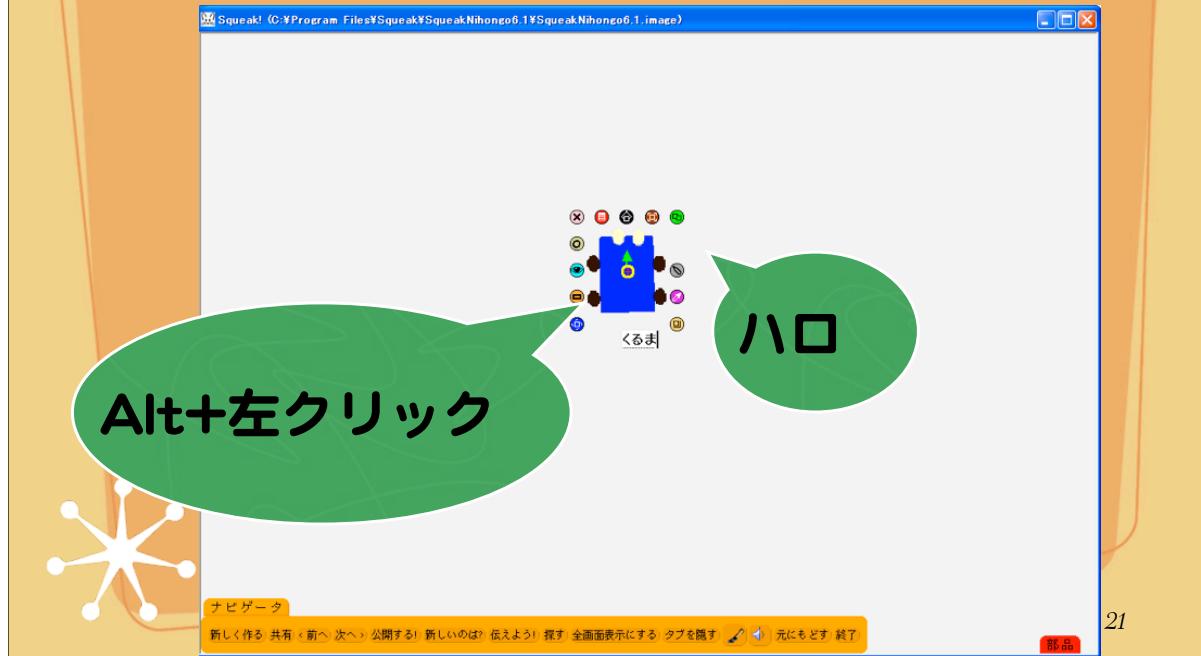
お絵描きツール



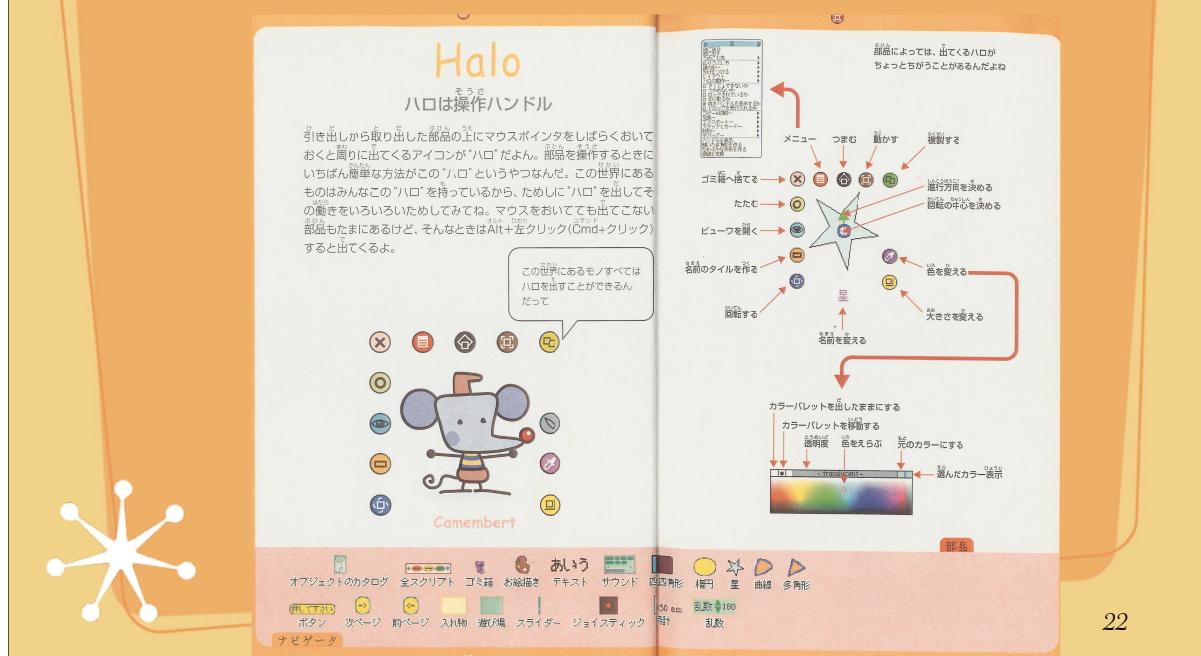
車ができました。



コントロールツール “ハロ”



コントロールツール “ハロ” の説明

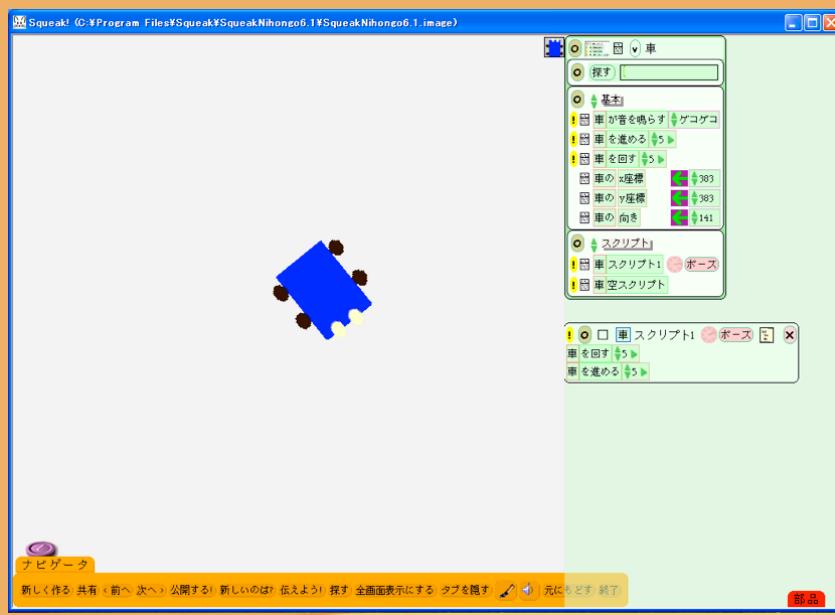


プロジェクト

- 車を作ろう
- 車を運転しよう
- 利口な車を作ろう

23

車を運転しよう



24

自分のビューワを開く

(2)
ここをクリック

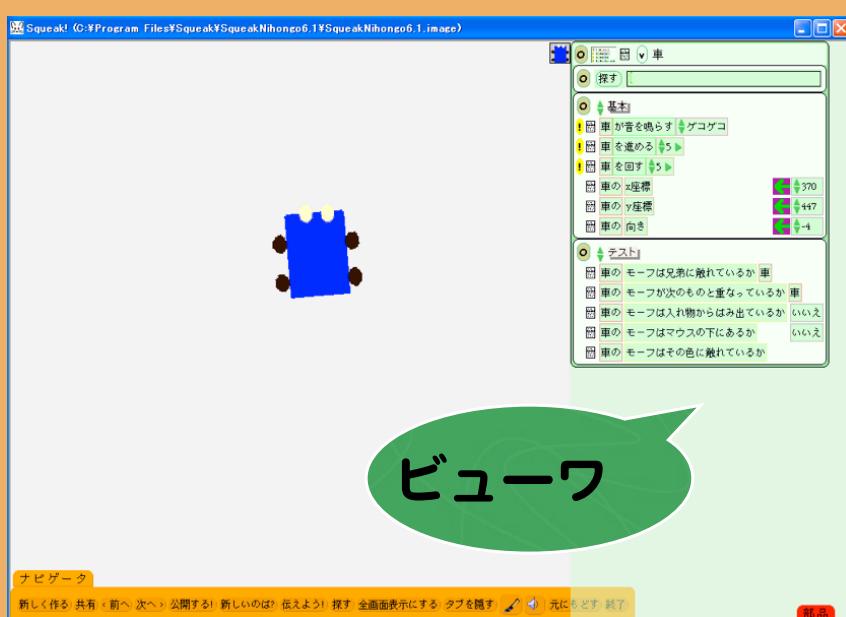


(1) Alt+左クリック

25

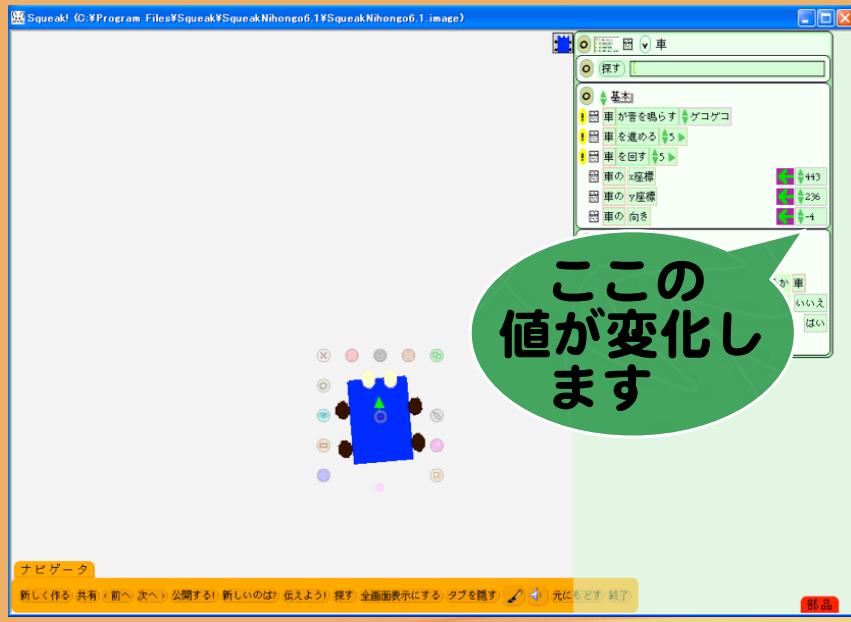
自分のビューワを開く

ビューワ



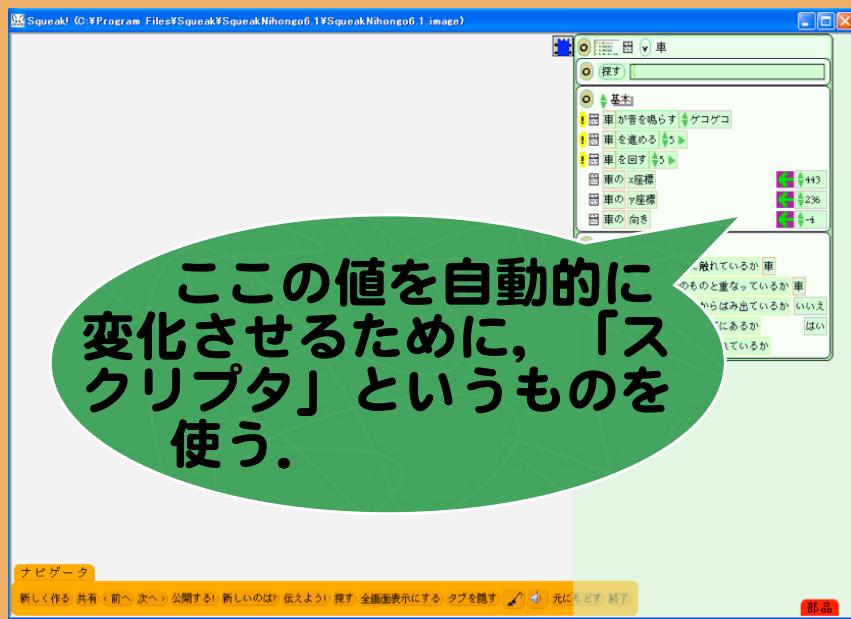
26

車を引っ張りまわすと...



27

車を自力で動かすには



28

スクリプタ

The screenshot shows the Scriptor software interface. On the left, a workspace displays a grid of tiles, with one tile highlighted in green and labeled "星の紋章". A speech bubble from a mouse character says, "それじゃあ、まずはピュータからタイルをひっぱり出してみよう。タイルがそのまま出でたり、要なのが出でたりするでしょ？ これはスクリプタじって、じゅもんの言物だよ。これはタイルを並べる所なんだ。はじめてタイルを出したときは自動でタイルも入った状態になって出てくるよ。" On the right, a script editor window titled "スクリプタ" shows a list of blocks. A red arrow points from the workspace to the script editor, with labels: "ピュータから出でてくる" (comes out from the computer), "タイルだけが出来てくる" (only tiles come out), and "スクリプタが自動で作られる" (the scripter is automatically generated). Another red arrow points from the script editor back to the workspace, with a label: "複数を実行しているときでも、タイルの入れ替えや背景の変更ができるんじゃよ" (Even when running multiple scripts, you can change tile types or backgrounds). A mouse character in the bottom right corner says, "これだけでじゅもんができるからなの？" (Is that all it takes to make a game?). The page number 29 is in the bottom right corner.

運転する手順を決める

The illustration shows a purple mouse character driving a small toy car. A speech bubble above the mouse says, "やっでもらいたい順番に並べるちゅ～！！" (Please arrange it in the order I want!!). To the right of the mouse, three green boxes list the steps: "前に進む" (Move forward), "回転する" (Turn), and "音を出す" (Make sound). A blue arrow points downwards from the last step towards a pink oval at the bottom. The page number 30 is in the bottom right corner.

スクリプタの表示



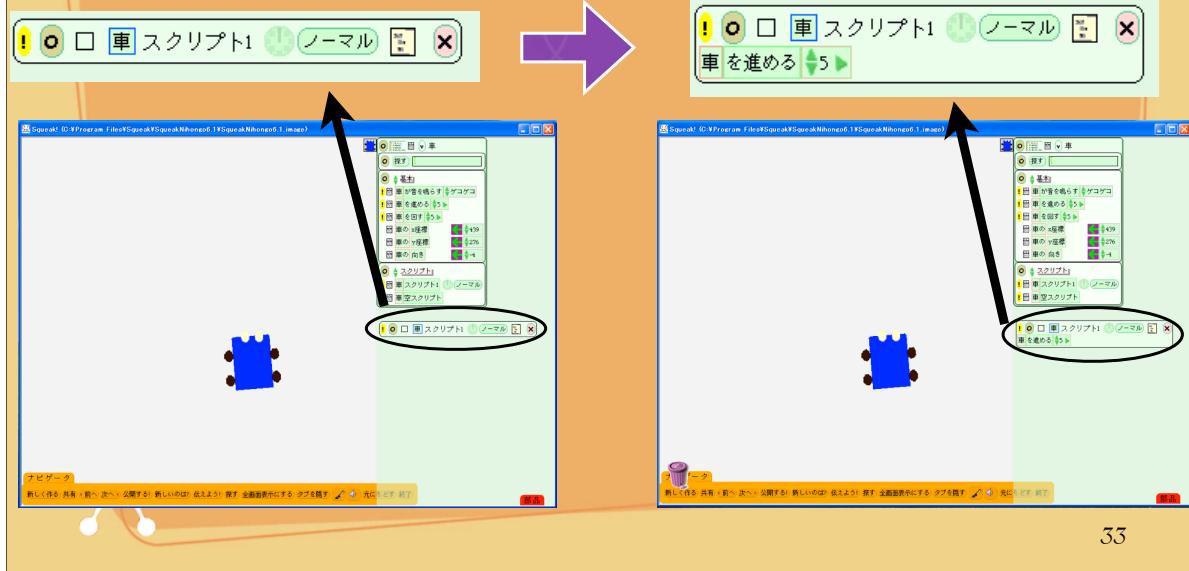
31

スクリプトの編集



32

スクリプトの編集の前後



33

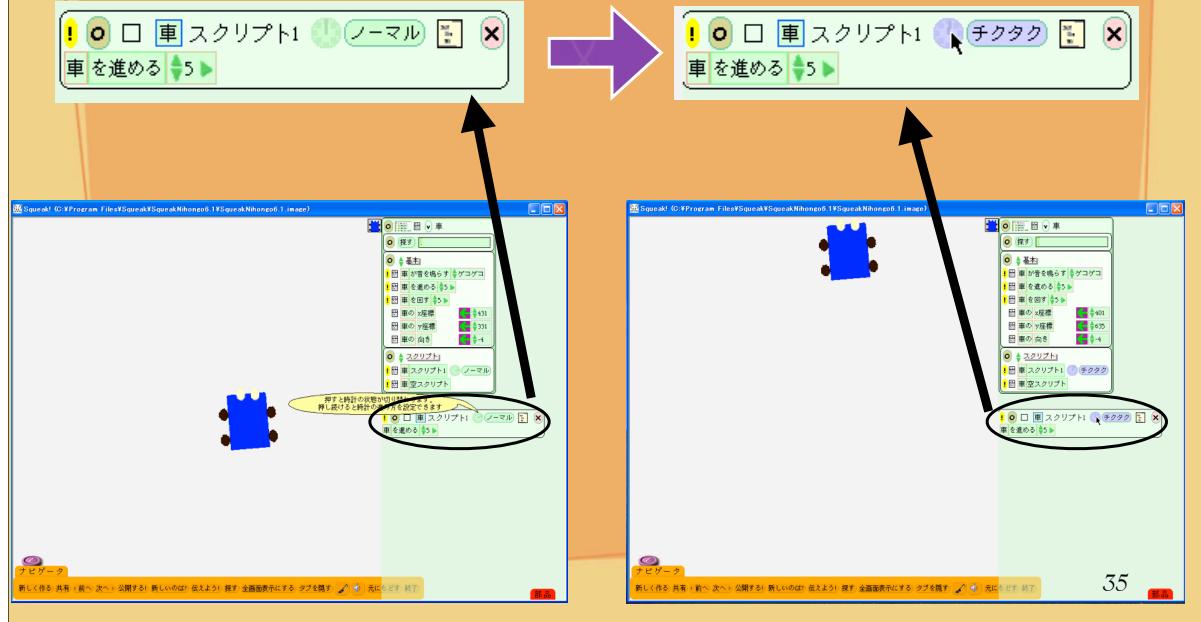
スクリプトを動かす



「時計の部分」をクリックすると、
スクリプトが動きます。

34

車を前進させる



35

車を回転させ前進させる



「時計の部分」を
クリックする。

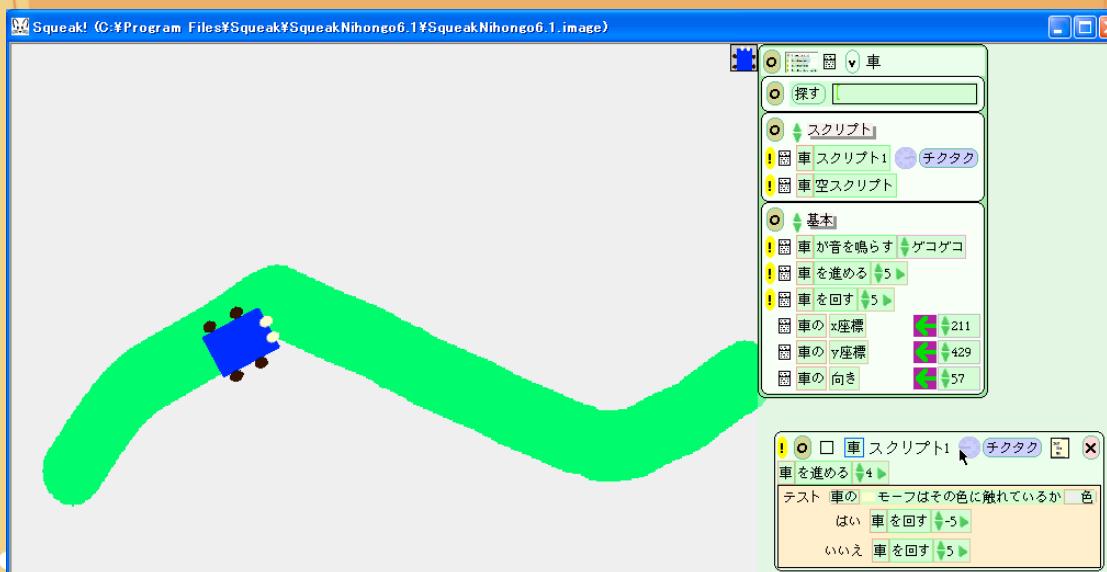
36

プロジェクト

- 車を作ろう
- 車を運転しよう
- 利口な車を作ろう

37

利口な車を作ろう



38

準備：コースを作る



コースに沿って車を走らせてみよう

- スクリプト1の「車を回す」の値を変化させて、車をコースに沿って走らせてみよう。
- コツ：「車を進める」の値を小さくする。



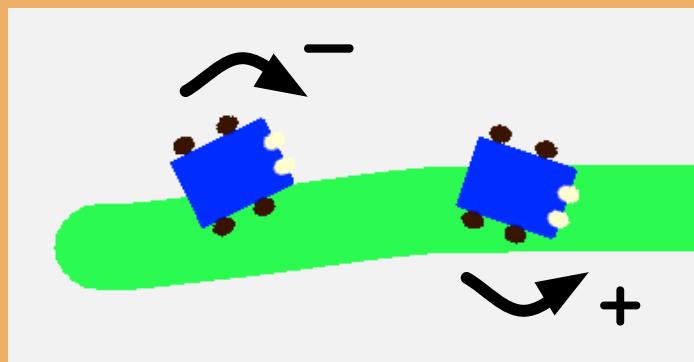
自動的にコースに沿って車を走らせてみよう

- コースから外れないように「自動的」に車の向きを調整する。
- 「車の向きを調整する」ためのルールを決める。

41

車の向きの調整ルール：1

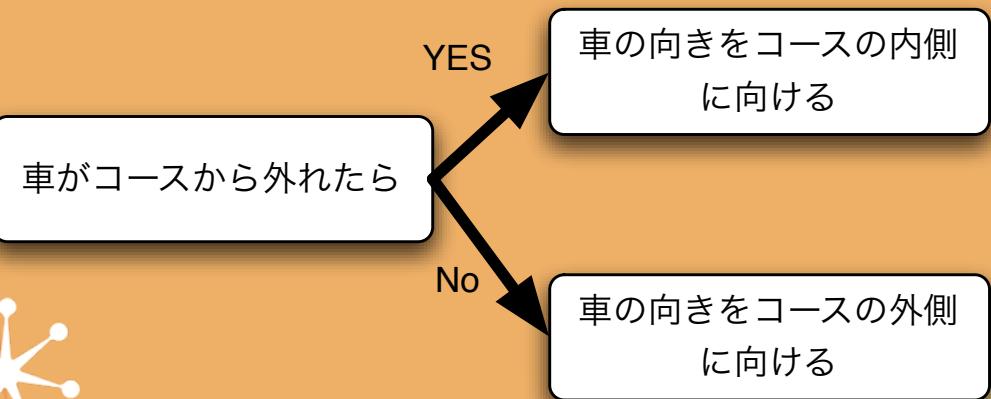
- 車がコースから外れたとき、車の向きをコースの内側に向けるようにする。



42

車の向きの調整ルール：2

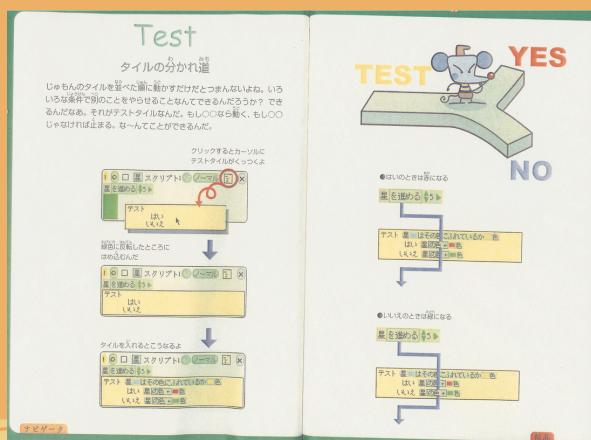
- 調整ルールは、「別れみち」になっている。



43

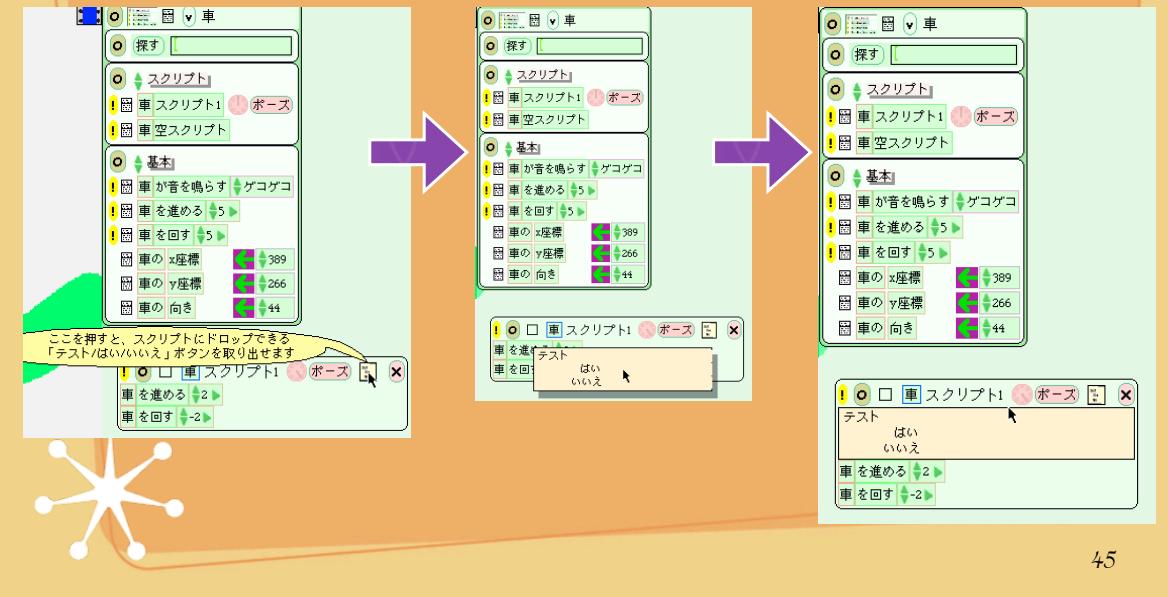
「Test (テスト)」

- 調整ルールを作るために、「テスト」というものを使う。



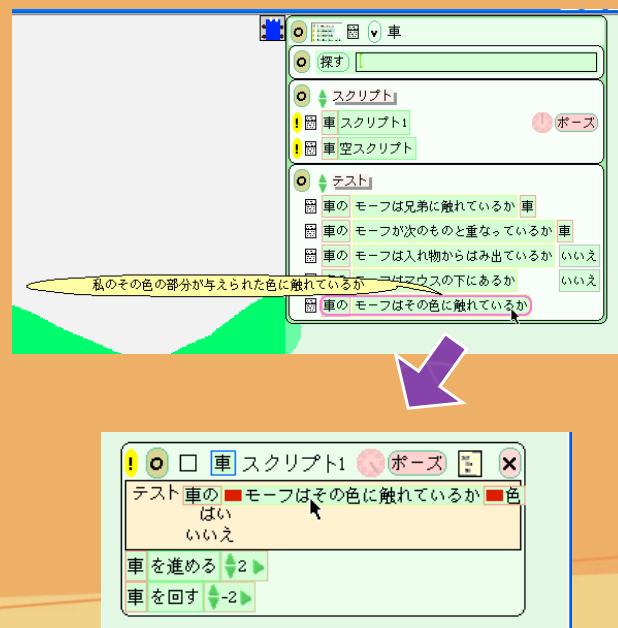
44

「テスト」の代入：1



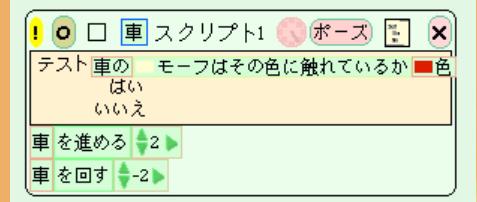
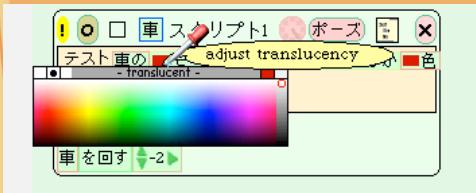
45

「テスト」の代入：2

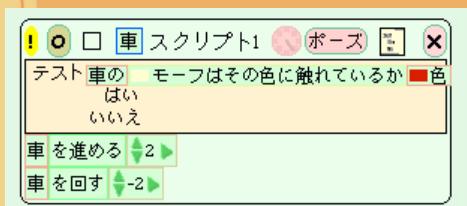


46

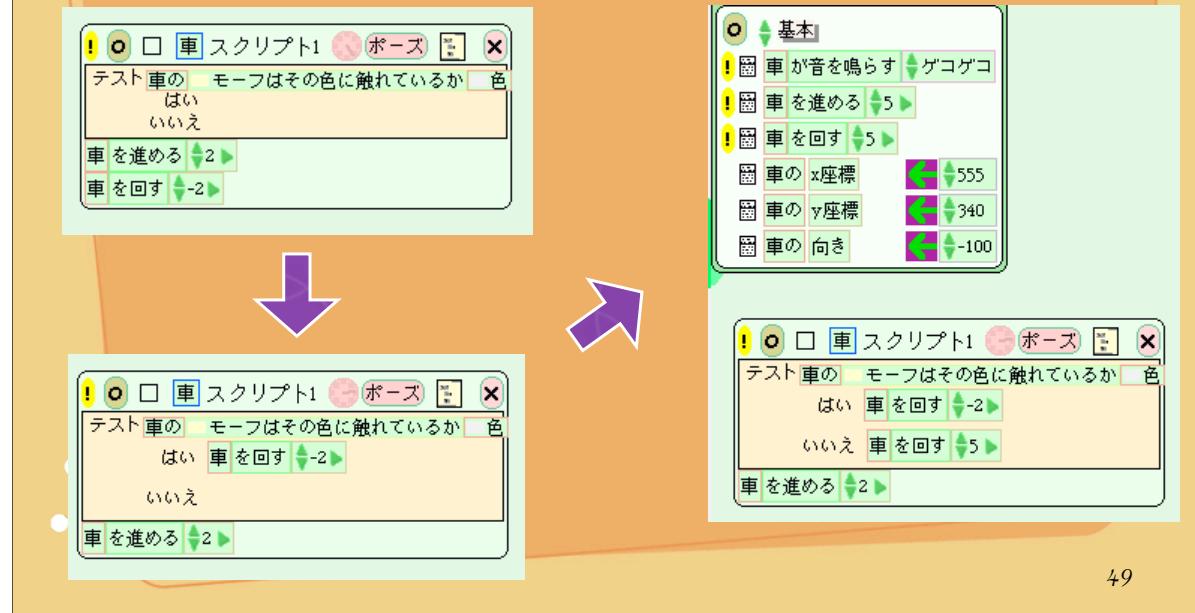
調整ルールの作成：1



調整ルールの作成：2

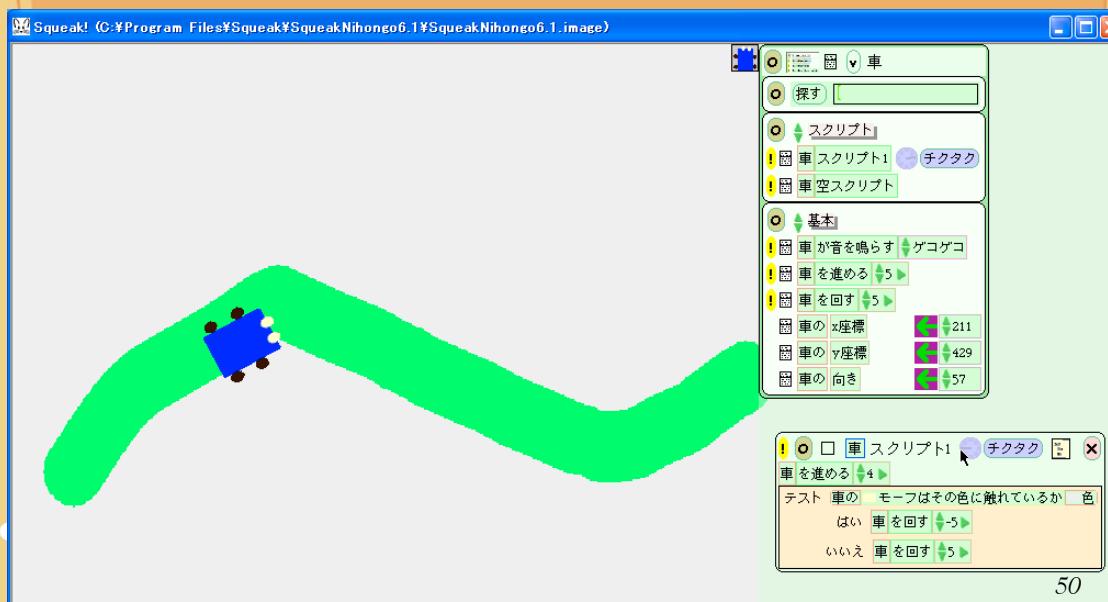


調整ルールの作成：3



49

調整ルールに基づいて、 自動車を動かす



50

自動制御： フィードバック制御

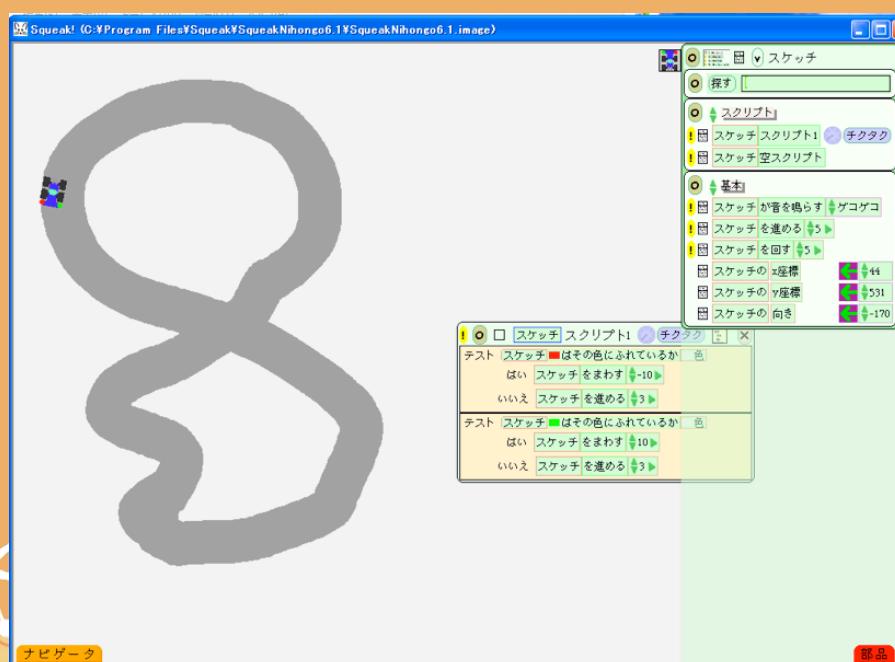
- 調整ルールに基づいて、何かを制御する。
- ここでは、車の方向を制御している。
- 例：クーラーの温度調整も同じ原理



詳しい内容は、大学の2年生ぐらいで
勉強する。

51

もっと利口な車



52

まとめ



- 「スクイーク」を使って、「コンピュータを操る方法（プログラミング）を実習しました。
- コンピュータの画面で、車を制御するソフトを作成した。



53

質問



?

■

hideaki@jaist.ac.jp



54