CTC Past Projects

Selected CTC 2016 & 2017 Projects

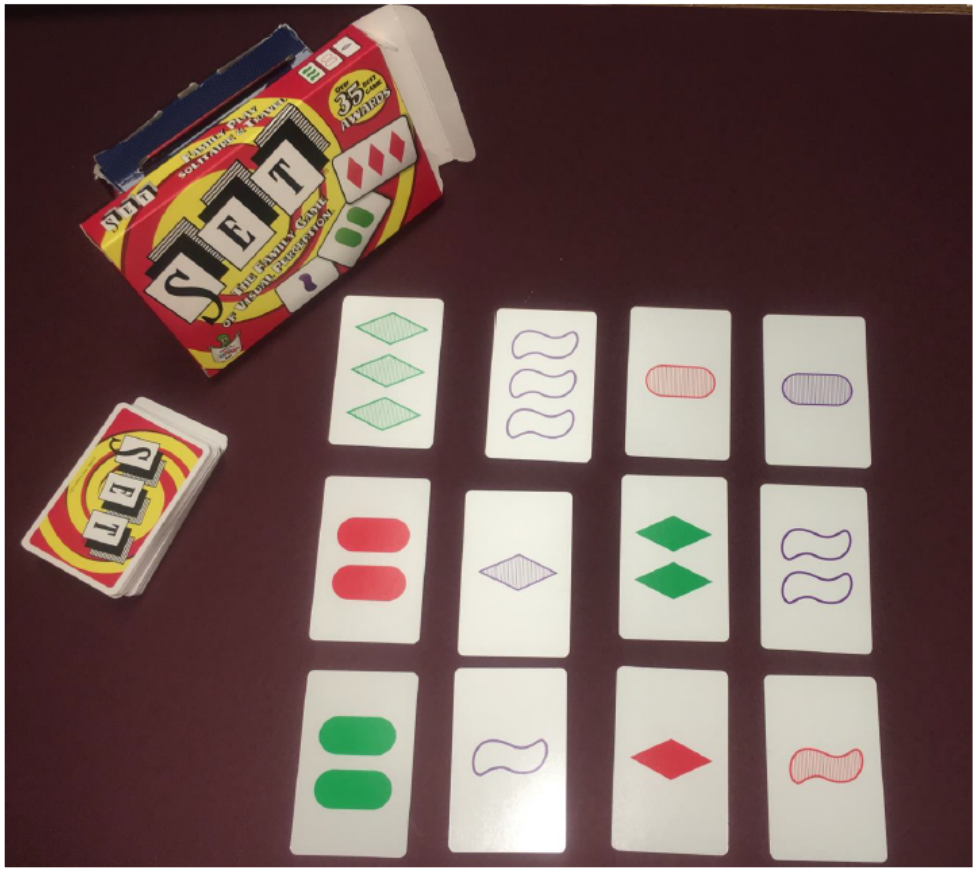
- Aquaponics Simulation
- The Maze Game
- Planets Orbit Simulator
- States of U.S.
- Minecraft Fireworks
- Persistence of Vision
- Scopophobia 2
- Stock Market Game
- Simulation of Chemical Reaction
- Automatic Downspout System
- Location Finder
- Set Trainer
- Minecraft 2D

Set Trainer

[CTC 2017 1st Prize Winner]

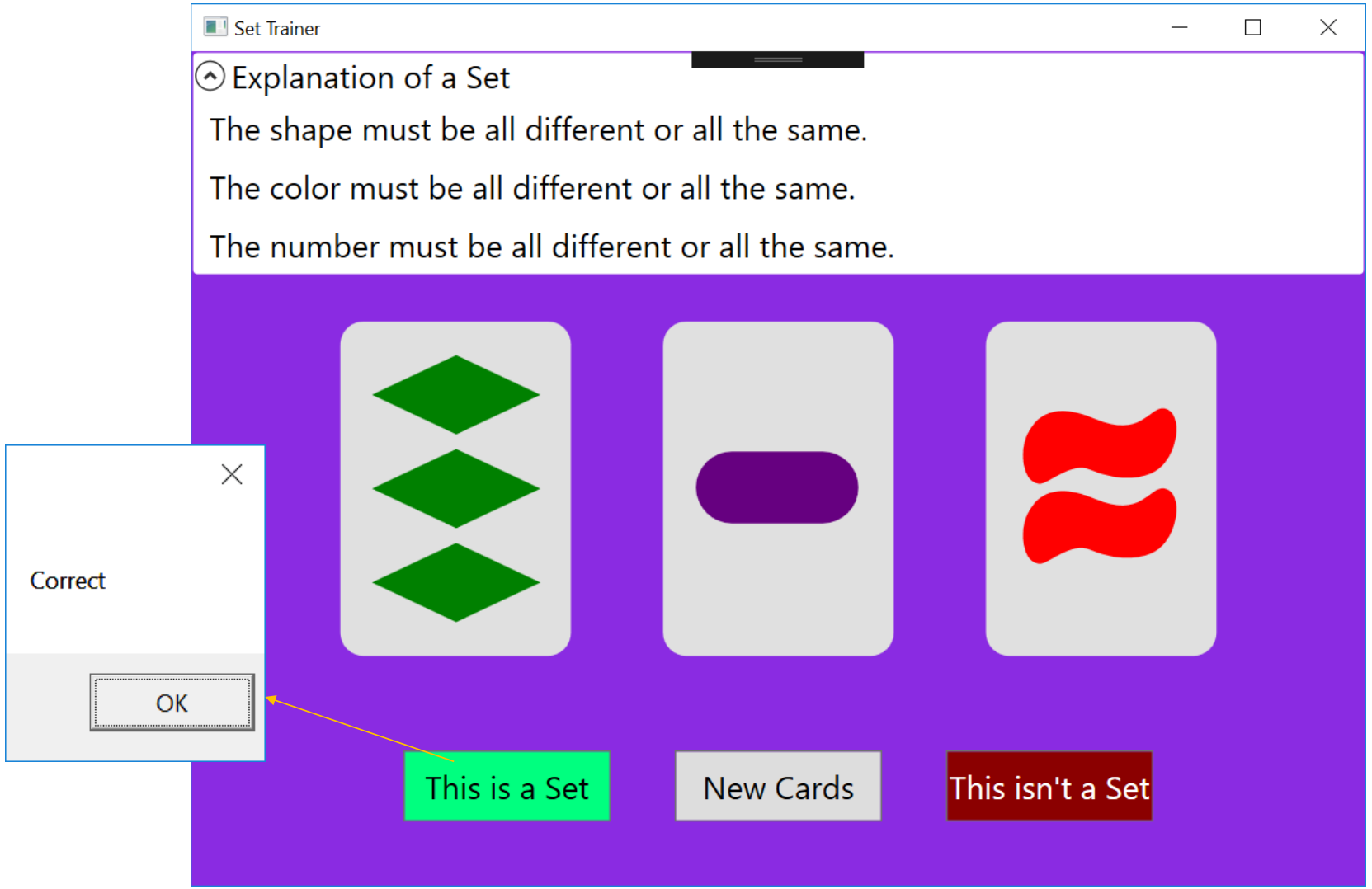
Motivation

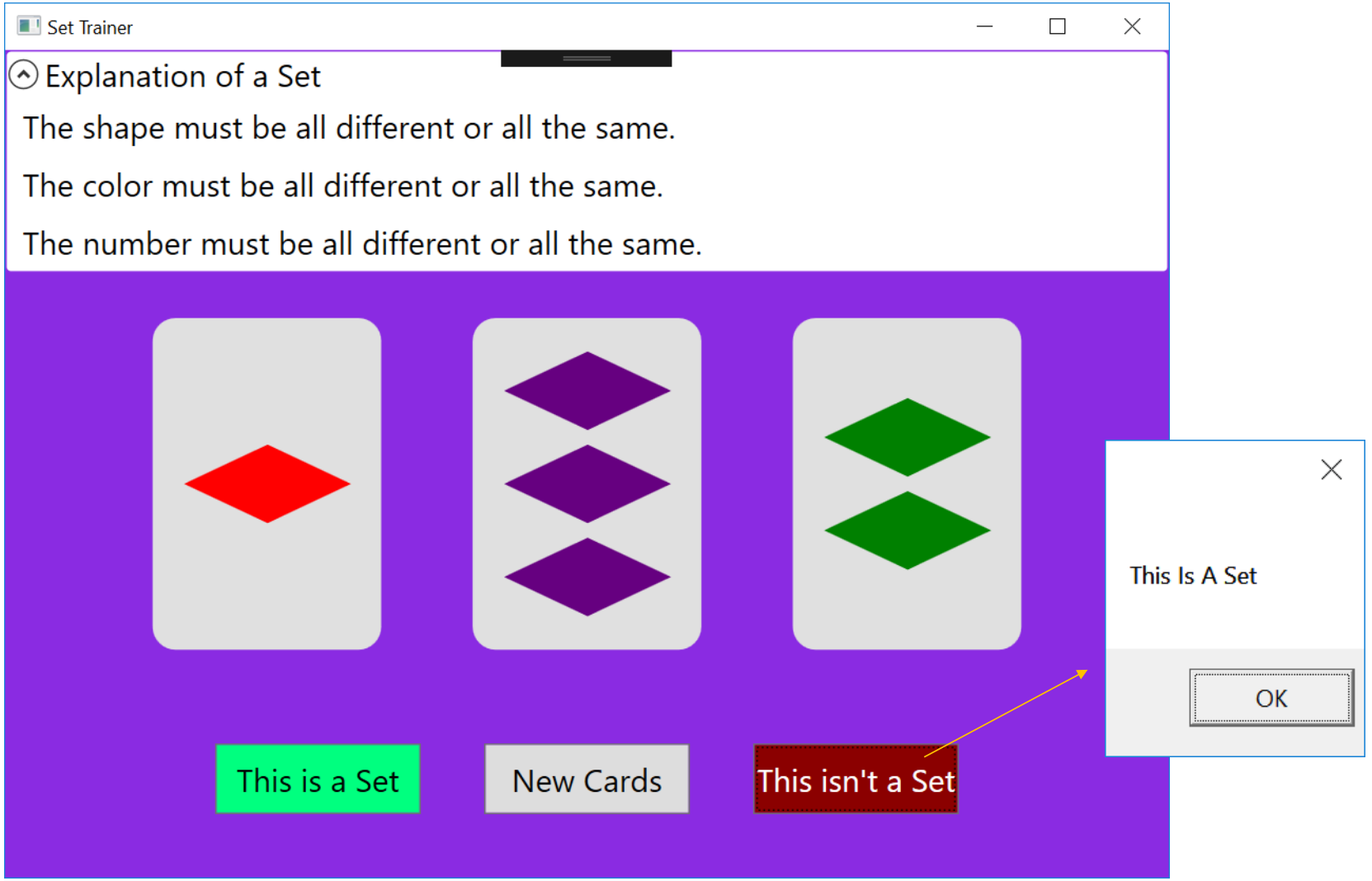
- Train people to play the card game “Set”.
- Help people recognize and understand what a set is.



Description

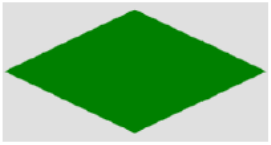
- “Set” is a matching game using 27 cards for the basic deck and 81 cards for the advanced deck.
- This program will help people recognize and understand what a set is.
- The user will see 3 cards displayed and then they will need to decide if it is a proper set. The program will tell them if they are right or wrong.
- The cards used in my program are only the “basic deck” cards, and not the complete advanced version.
- Program was written in C# and WPF using Visual Studio 2015 development environment.


Set Trainer
Explanation of a Set
The shape must be all different or all the same.
The color must be all different or all the same.
The number must be all different or all the same.
Correct
OK
This is a Set
New Cards
This isn't a Set


Set Trainer
Explanation of a Set
The shape must be all different or all the same.
The color must be all different or all the same.
The number must be all different or all the same.
This is a Set
New Cards
This isn't a Set
This Is A Set
OK

Details

- Scanned the images of squiggles, ovals, and diamonds, then put the images into Inkspace and made the shapes using the scanned imaged as a reference. Scanned images were converted from SVG to XAML.
- Below are XAML definitions for the different shapes of the squiggles, ovals, and diamonds. The “Data” numbers are the points and vectors that define the shape.



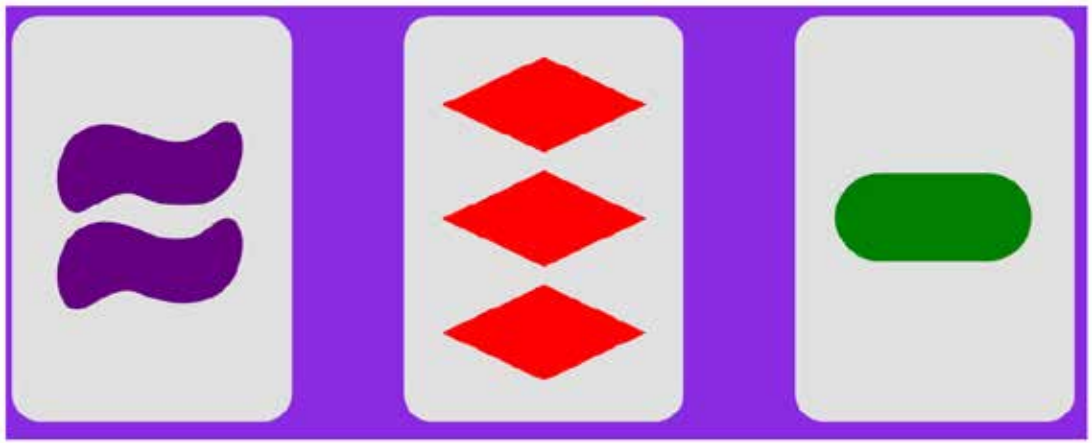
```
<Path Style="{StaticResource shape_Green} "Data="m 5,34 l68,-32 l68,32 l-68,32 z"/>
```

```
<Path Style="{StaticResource shape_Purple}" Data="m 33,62 a29,29 0 0 1 0,-58 l 77,0 a
29,29 0 0 1 0,58 z"/>
```

```
<Path Style="{StaticResource shape_Red}" Data="m 10,25 c 7.45106 -18.99865 24.10539 -
28.90139 56.43506 -14.977 40.21455 17.32041 47.83211 -10.36829 59.47387 -8.68231
14.68848 2.1272 11.00121 39.56756 -8.68231 50.79155 -7.97875 4.54967 -26.42469 9.8015
-53.17919 -1.5194 -18.13537 -7.6738 -37.53305 12.77552 -44.49688 11.72112 -13.60411 -
2.05981 -13.84716 -26.3785 -9.55055 -37.33396 z"/>
```

Details (cont.)

- All twenty-seven cards using the three basic shapes were made.



- Labeled cards were put into a list.

```
CardList.Add(new Card("Diamond","Green",1,(Style)TryFindResource("GreenDiamond1")));
CardList.Add(new Card("Diamond","Green",2,(Style)TryFindResource("GreenDiamond2")));
CardList.Add(new Card("Diamond","Green",3,(Style)TryFindResource("GreenDiamond3")));
CardList.Add(new Card("Oval", "Green", 1, (Style)TryFindResource("GreenOval1")));
CardList.Add(new Card("Oval", "Green", 2, (Style)TryFindResource("GreenOval2")));

etc…
```

Details (cont.)

- Once the cards were made, “if-else” statements were created to check what is and isn’t a “set”. There are three things that need to be checked:
  1. Checking that the shapes on the three cards are all the same OR if they are all different.
  2. Checking that the color on the three cards are all the same OR if they are all different.
  3. Checking that the number of shapes on the three cards are all the same OR if they are all different.
- If any of the statements above are FALSE, then it is not a “set”.
- Program shows “non-sets” 60% of the time, and 40% of the time it shows “sets”, to make it like the actual game.