

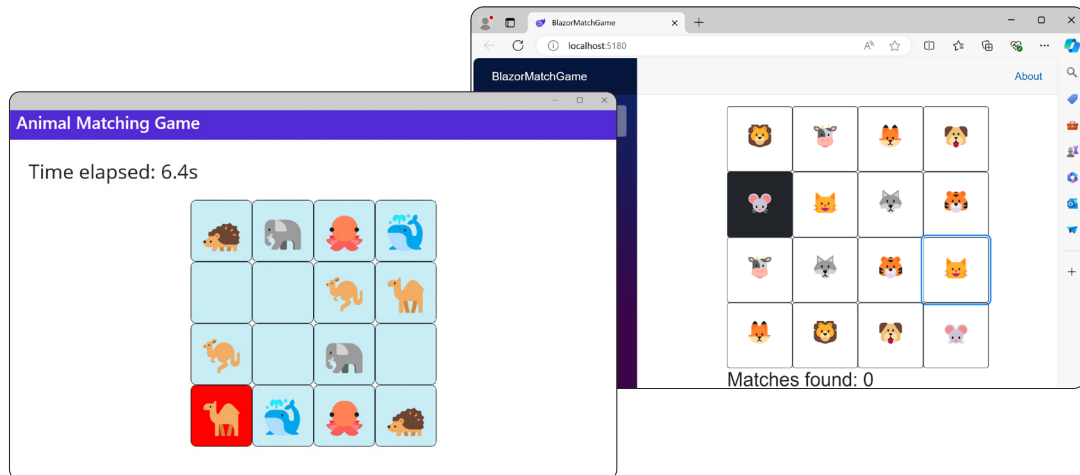
PDF



Downloadable exercise: Animal match boss battle

If you've played a lot of video games (and we're pretty sure that you have!), then you've had to play through a whole lot of boss battles—those fights at the end of a level or section where you face off against an opponent that's bigger and stronger than what you've seen so far. We have one last challenge for you before the end of the book—consider it the *Head First C#* boss battle.

In Chapter 1 you built an animal matching game. It was a great start, but it's missing... something. Can you figure out how to turn your animal matching game into a memory game? Go to our GitHub page and download the PDF for this project—or if you want to play this boss battle in Hard mode, just dive right in and see if you can do it on your own.



The rules of the boss battle

Your job is to turn your animal matching game from Chapter 1 into a memory game. Here's how it will work:

1. When the game starts, all of the animals are hidden. The timer works exactly like it did in Chapter 1.
2. The player clicks pairs of hidden animals. When the first hidden animal is clicked, the game reveals it.
3. When player clicks the second animal, if it's a match then both animals stay displayed. If it's not a match, then both animals are hidden again.
4. The game is over when all pairs have been found and displayed. At the end of the game, all of the animals will be displayed.
5. When the player starts a new game, the animals are shuffled and hidden again.

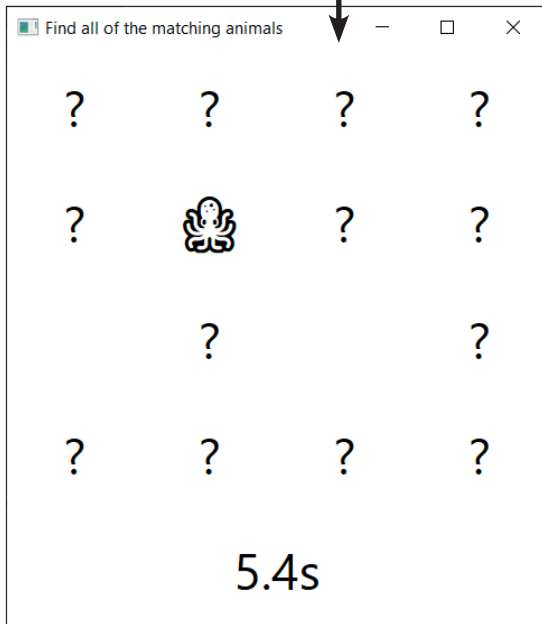
This is a project for you to do on your own!

If you go to the GitHub page for the book, you won't see any code for this project, because we want to give you the freedom to tackle this project in any way that you see fit.

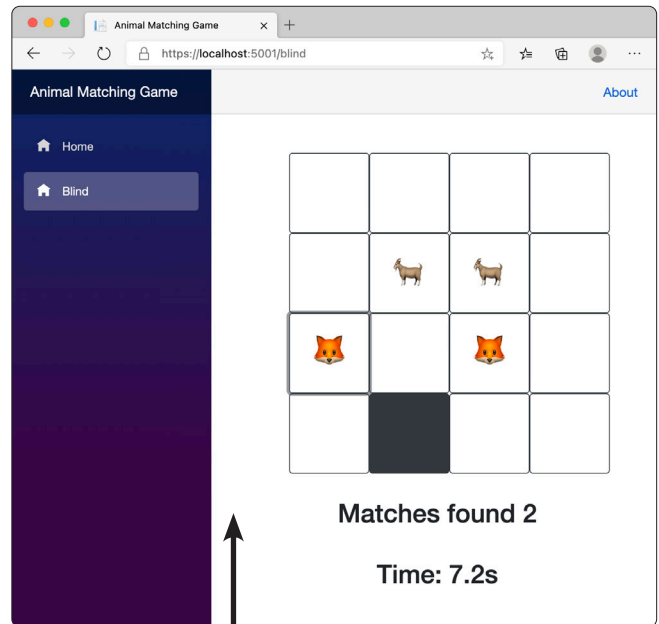
However...

Sometimes it really helps to have some hints. Luckily, we have some very creative readers! Here are two of the great solutions we've seen.

Here's a really innovative version of the game from one of our 4th edition readers. GitHub user [charl4ee](https://github.com/charl4ee/ImageMatch) turned the WPF version of the game into a memory game, and also adding a great effect where unmatched animals are displayed for an extra second before disappearing. You can get the source code here: <https://github.com/charl4ee/ImageMatch>



↑
The 4th edition of the book used a different technology, WPF, that uses XAML to create Windows desktop apps, which is why this screenshot looks a little different.



Here's an Blazor version from another one of our readers. GitHub user Pan (bodow) made some significant improvements to the Blazor version of the game. In the default mode, it keeps track of the high score, encouraging you to beat your best time. And more importantly, they also added a "memory" version where the animals are hidden, and only appear when you click them. You can get the source code here: <https://github.com/bodow/BlazorMatchGame>

If you come up with your own innovative solution, we'd love to see it! Publish the code to GitHub and get in touch with us on our GitHub page, YouTube channel, or social media.

Tips for tackling the challenge

Doing this project on your own is a great way to kickstart the next stage of your learning. While we want you to **solve this on your own**, here are some things to think about that might help you along the way.

Think About State

How will you keep track of which animals are hidden and which are revealed?

What information does your game need to remember between clicks?

When do you need to update the state of your game?

Break It Down

Try tackling **one new feature at a time**.

Start by figuring out how to hide all animals at the beginning...

...then work on revealing animals when clicked...

...and finally, implement the logic for matching pairs.

Hmm... Every time a button gets clicked, I need to think about what should happen next. Does it reveal an animal? Hide animals? Check for a match? There's a lot to think about!

Debug Strategically

Use `Debug.WriteLine` or the debugger to track game state.

Add temporary visual indicators while testing.

Test each new feature thoroughly before moving on.

Some Pitfalls to Avoid

Don't try to change everything at once.

Remember to handle **edge cases** (like clicking the same animal twice).

Make sure your timer still works with the new game mechanics.

Make it Look Good!

Consider using button background colors or opacity to “hide” animals.

Think about how to visually indicate selected animals.

Plan how to show/hide animals smoothly.

Remember: There's no single “right” solution. Focus on making your code clear and easy to maintain. If you get stuck, try breaking the problem down into smaller pieces.



Get creative!

We started the book with the animal matching game because it's a great platform for you to get creative. There are so many “even-better-ifs” that you can try out. Here are a few ideas:

- ★ Can you figure out how to add rows or columns to your game?
- ★ Create a version where the player has to match three animals, not two.
- ★ Make the timer count down, not up. Add extra rounds with shorter timers.
- ★ Instead of matching animals, try matching pairs of related icons (like a fish 🐟 and a fishing rod 🎣).
- ★ Make the game more challenging by hiding some of the pairs the player previously found if too much time has elapsed.


Looking for more inspiration? Check out this blog on the Visual Studio Blog written by one of our authors, Andrew Stellman, that highlights creative work by other *Head First C#* readers from the 4th edition: <https://devblogs.microsoft.com/visualstudio/head-first-csharp-contest-got-some-truly-creative-entries/>

Did you come up with a creative or interesting version of the animal matching game? If you did, then we want to hear from you! Publish your code to GitHub, then reach out to us through GitHub or on social media. We may even include it in this PDF to help provide inspiration and encouragement to other readers.

And finally... thank you for reading our book!

It means so much to us that we've helped you on your C# learning journey. If you feel like this book helped you, we hope you consider leaving a five-star review for our book on Amazon:

<https://www.amazon.com/Head-First-CSharp/dp/1098141784/>



Even a short, one-sentence review with five stars makes a big difference in helping us reach more readers, so we'd really appreciate it! (Oddly enough, four-star reviews can hurt our rankings a bit.) We love our readers, and we're grateful for your time!

— Jenny and Andrew