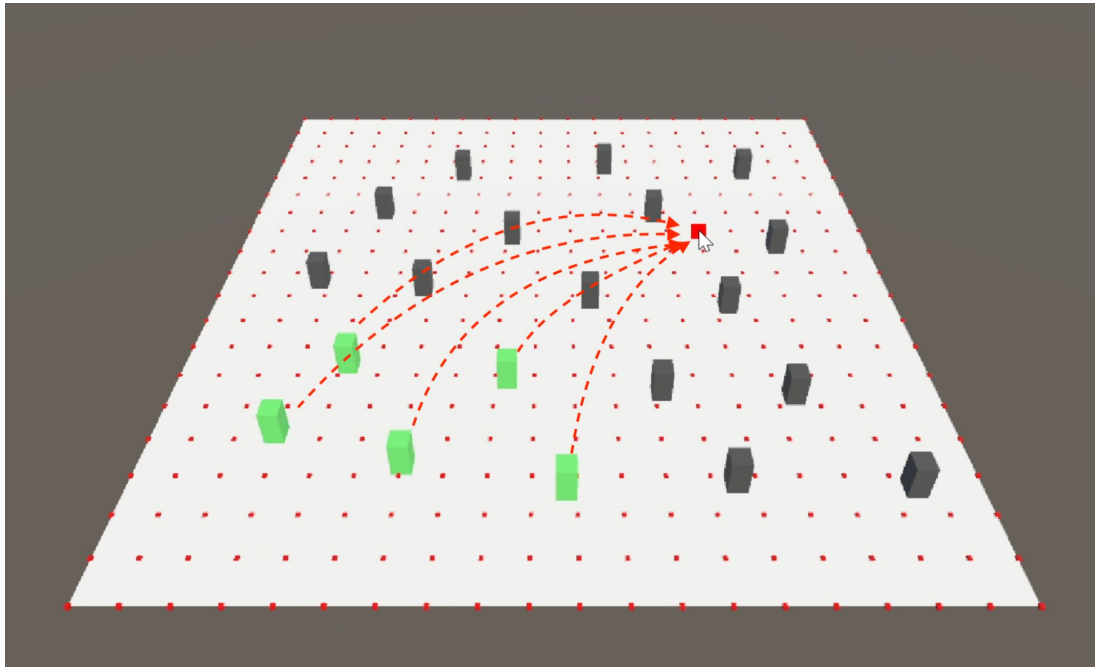
AI Navigation
A* Pathfinding Algorithm

In Unity



How does A* work?

1.) Start at the root.
2.) Find a neighbouring node with the smallest `f` score where
`f(n) = g(n) + h(n)`

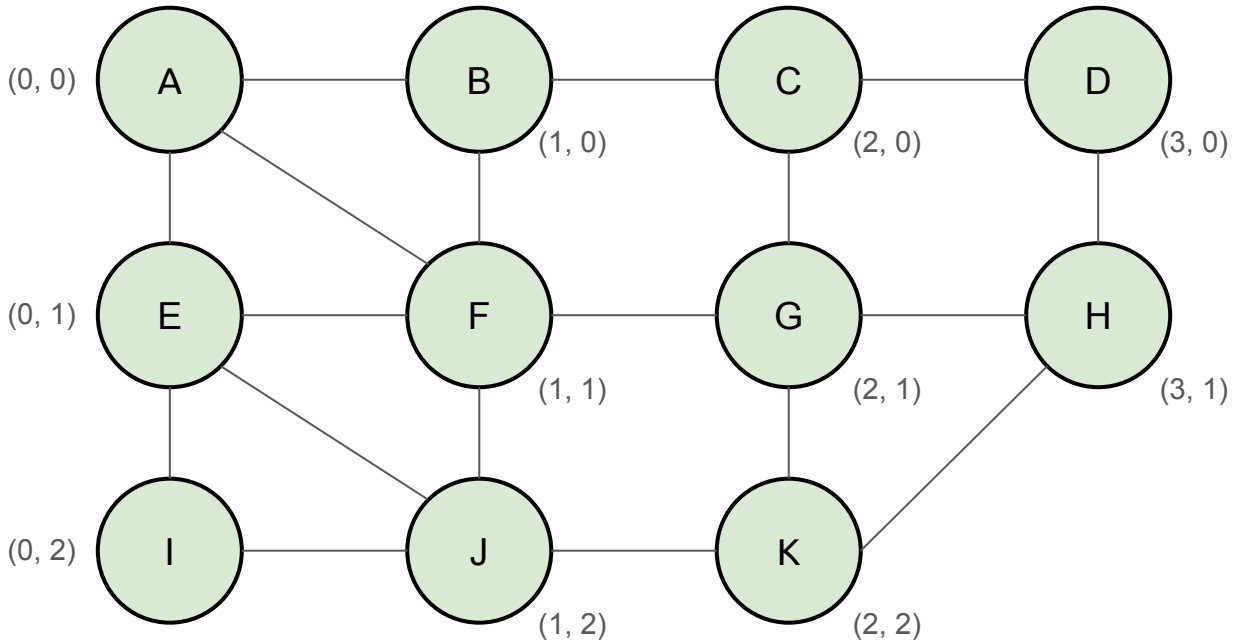
`g(n)`: cost from start to current node

`h(n)`: cost from current node to end node

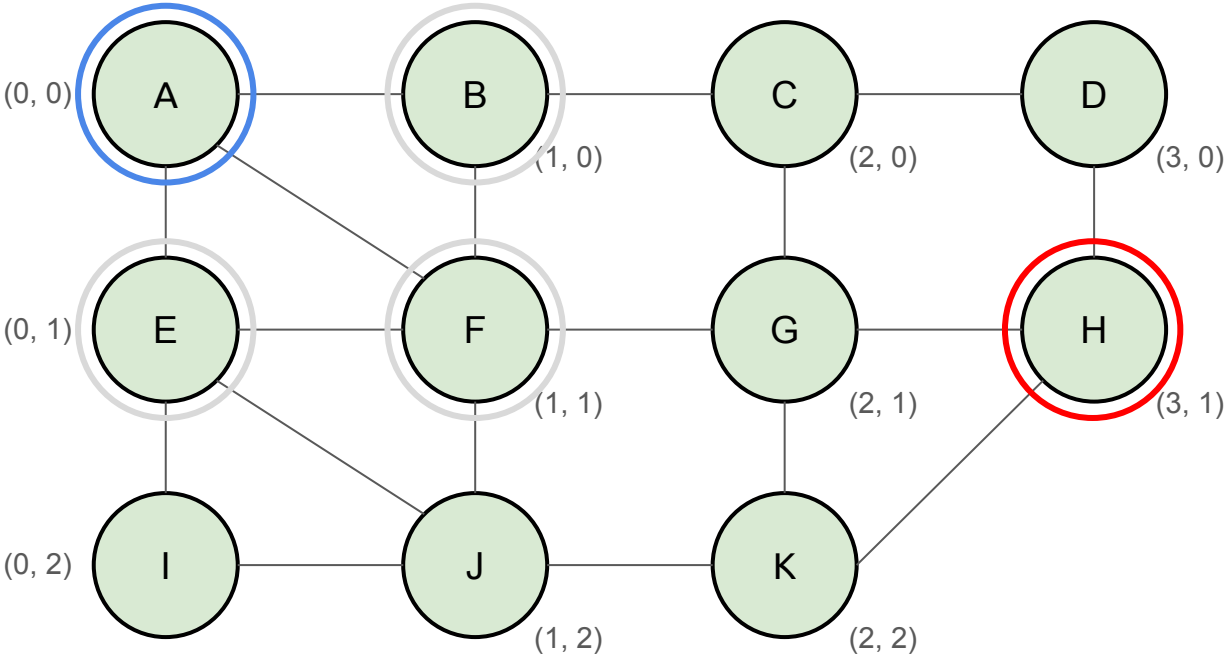
3.) Visit the node and repeat until the end node is found or all nodes have been explored.

Visualizing A*

g(n): from start to the current node.
h(n): from current node to the end node.
f(n) = g(n) + h(n)



Visualizing A*: A to H



$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}.$$

g(n): from start to the current node.
h(n): from current node to the end node.
f(n) = g(n) + h(n)

dist(i, j): euclidean distance
start node: **A**
end node: **H**
current node: **A**

A to A
g((0,0)) = dist(A, A) = 0
h((0,0)) = dist(A, H) = sqrt(8) ~= 2.83
f((0,0)) = 0 + sqrt(8) = sqrt(8) ~= 2.83
neighbours: B, F, E

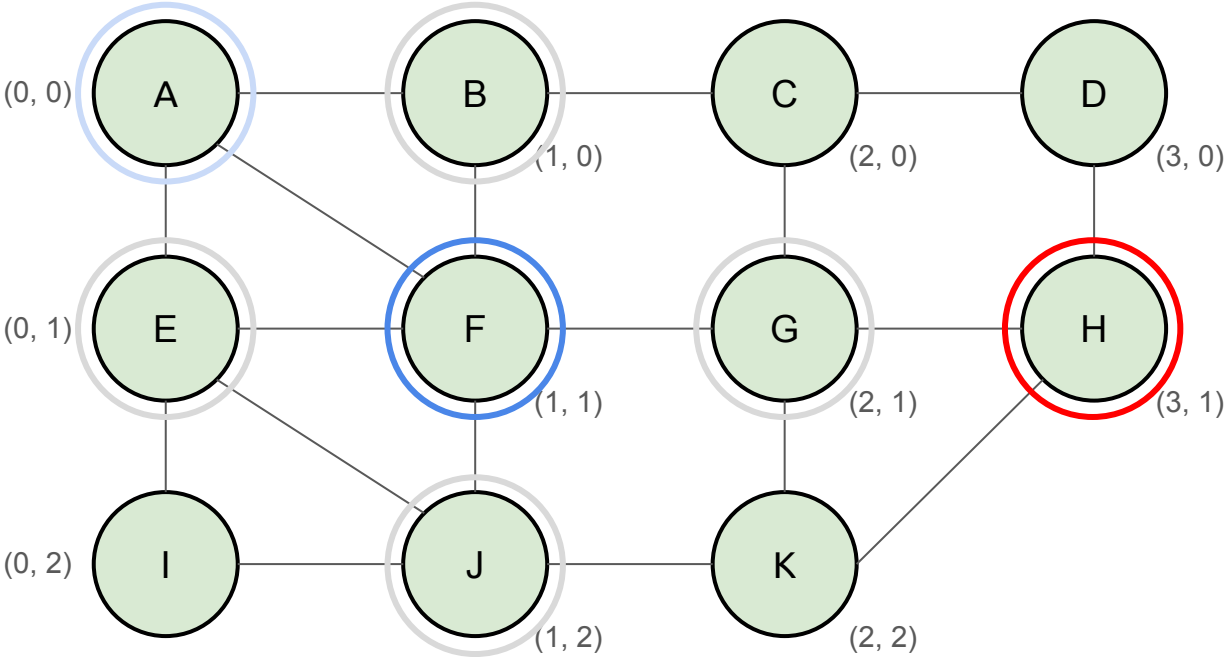
A to B:
g((1,0)) = dist(A, B) = 1
h((1,0)) = dist(B, H) = sqrt(5)
f((1,0)) = 1 + sqrt(5) ~= 3.24

A to F:
g((1,1)) = dist(A, F) = sqrt(1) = 1
h((1,1)) = dist(F, H) = 2
f((1,1)) = 1 + 2 = 3

A to E:
g((0,1)) = dist(A, E) = sqrt(1) = 1
h((0,1)) = dist(E, H) = 3
f((0,1)) = 1 + 3 = 4

Choose lowest F score. **(F)**

Visualizing A*: A to H



$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}.$$

g(n): from start to the current node.
h(n): from current node to the end node.
f(n) = g(n) + h(n)

dist(i, j): euclidean distance
start node: **A**
end node: **H**
current node: **F**

F to B:
g((1,0)) = dist(F, B) = 1
h((1,0)) = dist(B, H) = sqrt(5)
f((1,0)) = 1 + sqrt(5) ~= 3.24

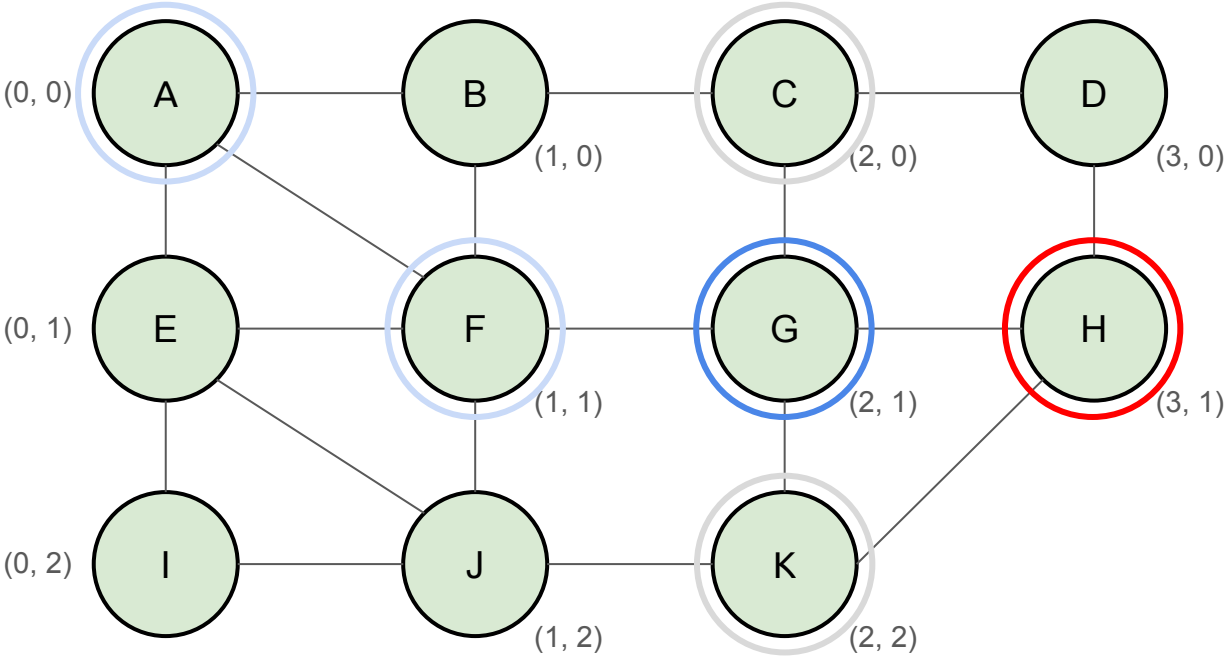
F to G:
g((2,1)) = dist(F, G) = 1
h((2,1)) = dist(G, H) = 1
f((2,1)) = 1 + 1 = 2

F to J:
g((1,2)) = dist(F, J) = 1
h((1,2)) = dist(J, H) = sqrt(5)
f((1,2)) = 1 + sqrt(5) ~= 3.24

F to E:
g((0,1)) = dist(F, E) = 1
h((0,1)) = dist(E, H) = 3
f((0,1)) = 1 + 3 = 4

Ignore A (already visited)
Choose lowest F score. **(G)**

Visualizing A*: A to H



$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}.$$

g(n): from start to the current node.
h(n): from current node to the end node.
f(n) = g(n) + h(n)

dist(i, j): euclidean distance
start node: **A**
end node: **H**
current node: **G**

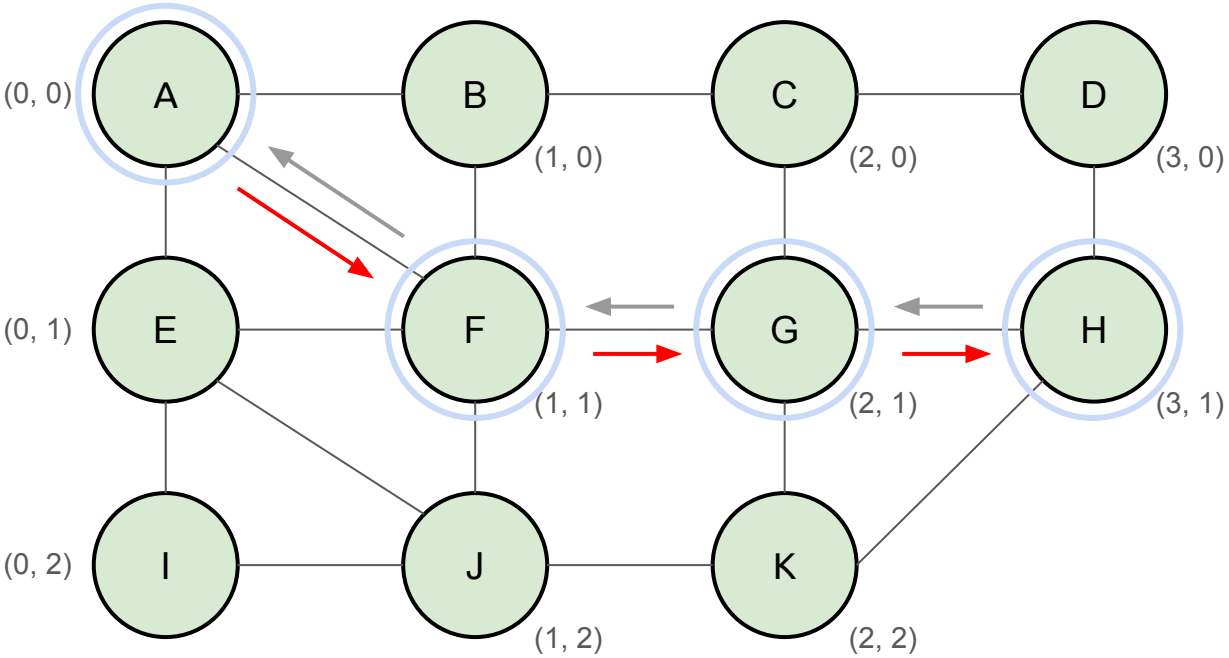
G to C:
g((2,0)) = dist(G, C) = 1
h((2,0)) = dist(C, H) = sqrt(2)
f((2,0)) = 1 + sqrt(2) ~= 2.41

G to K:
g((2,2)) = dist(G, K) = 1
h((2,2)) = dist(K, H) = sqrt(2)
f((2,2)) = 1 + sqrt(2) ~= 2.41

G to H:
g((3,1)) = dist(G, H) = 1
h((3,1)) = dist(H, H) = 0
f((3,1)) = 1 + 0 = 1

Ignore F (already visited)
Choose lowest F score. **(H and it is the goal.)**
Stop.

Visualizing A*: A to H



Reconstruct path by iterating through **cameFrom** node reference.

So, we get H -> G -> F -> A then reverse the list to get path from A to H:
A -> F -> G -> H