An Introduction to Data Structures

The LinkedList

The Linked List - LinkedList Basics

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

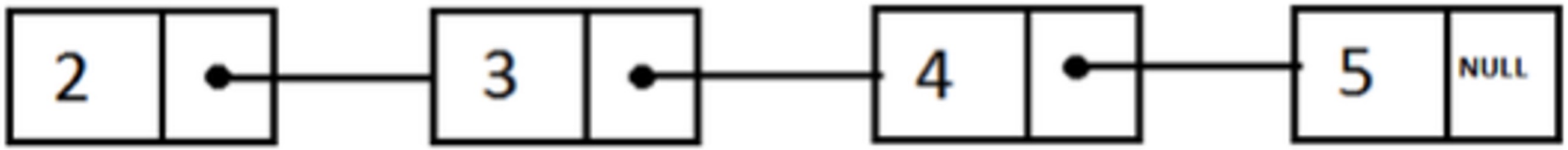
- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

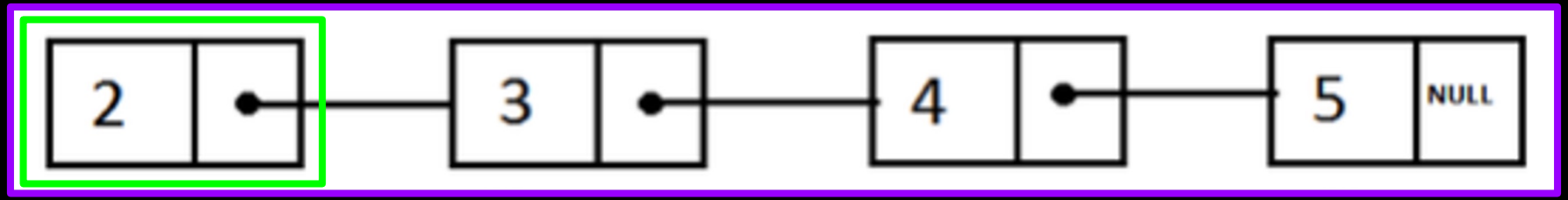
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



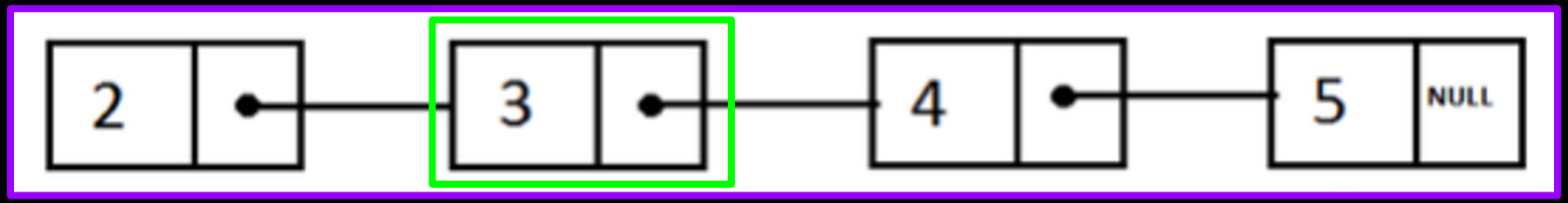
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



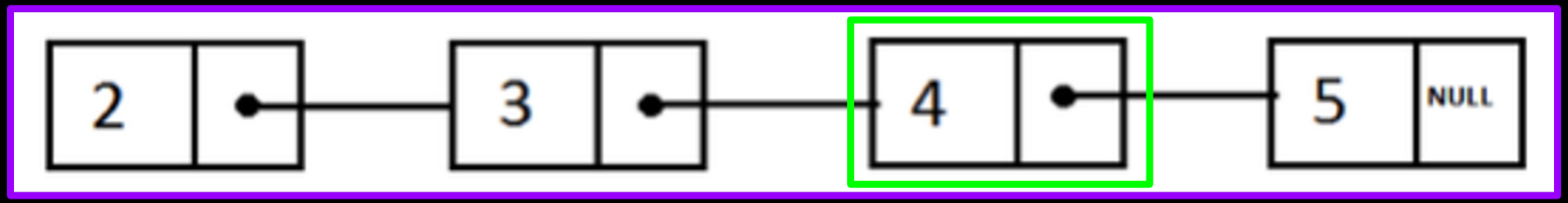
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



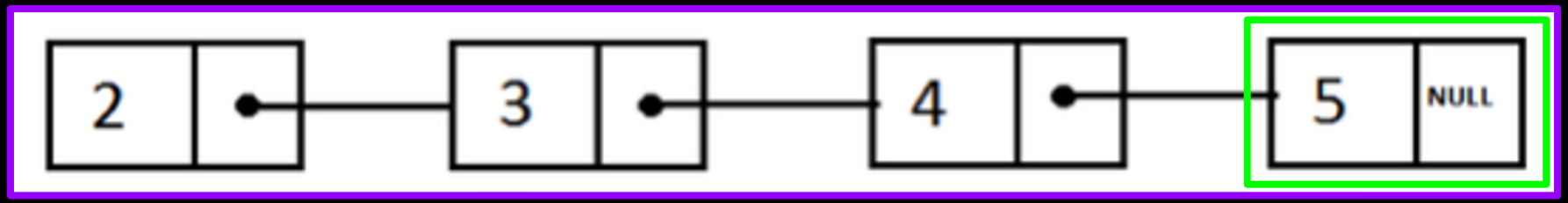
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



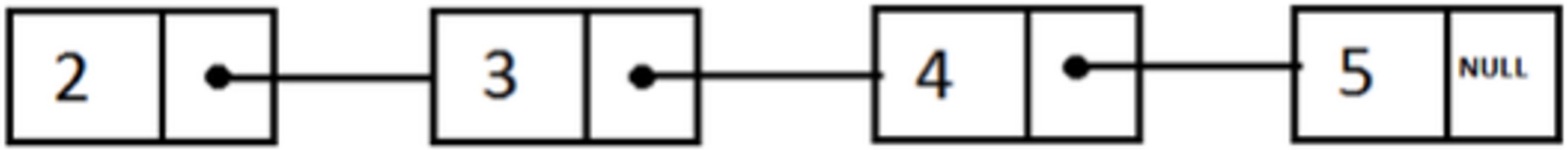
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

Salaries	10,000	12,500	8,750	15,000

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

Salaries	10,000	12,500	8,750	15,000

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

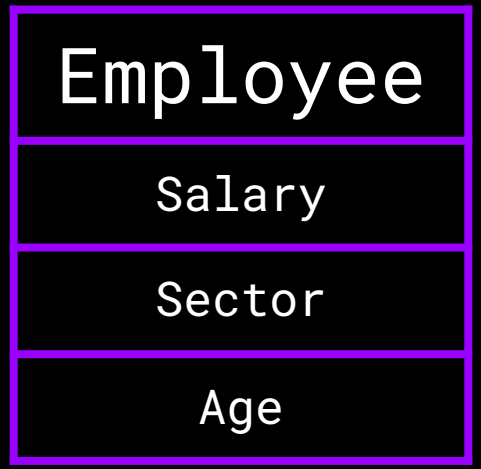
Salaries	10,000	12,500	8,750	15,000

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A **LinkedList** is a **sequential access** linear data structure in which every element is a separate **object** called a **Node**, which has 2 parts
  - The **data**
  - The **reference** (or pointer) which points to the next **Node** in the List

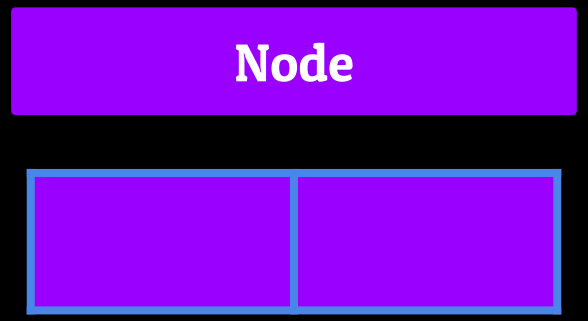
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

Node

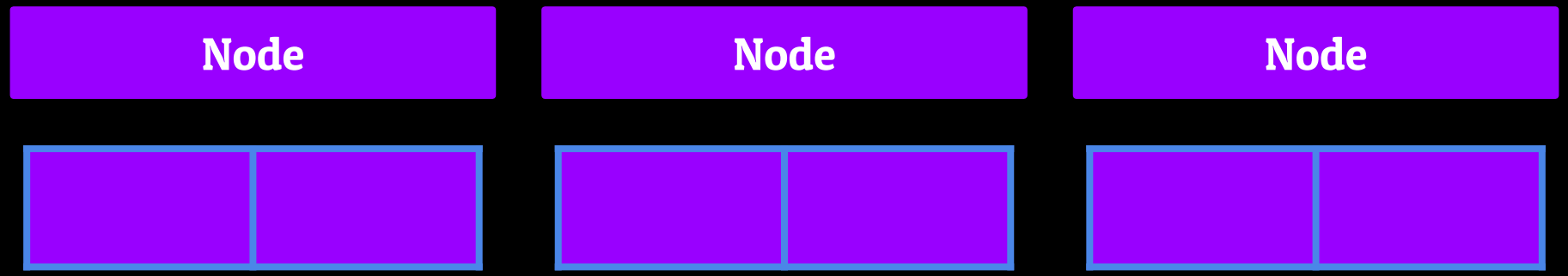
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



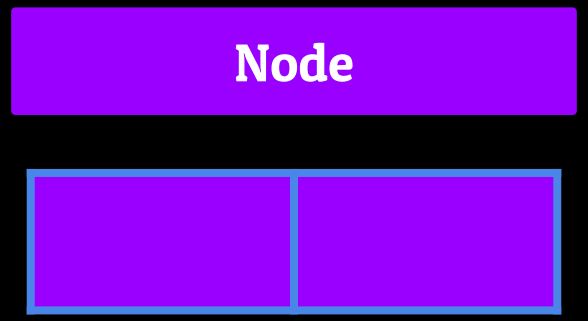
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



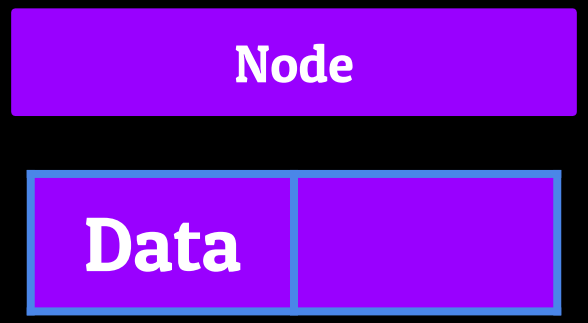
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



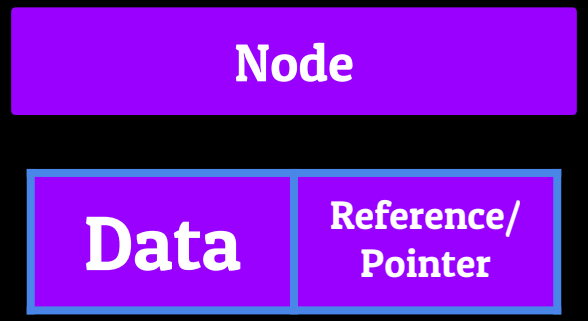
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



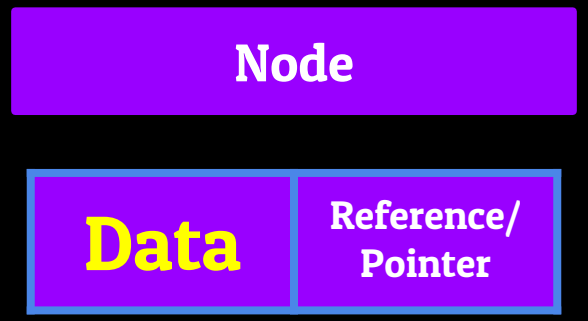
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



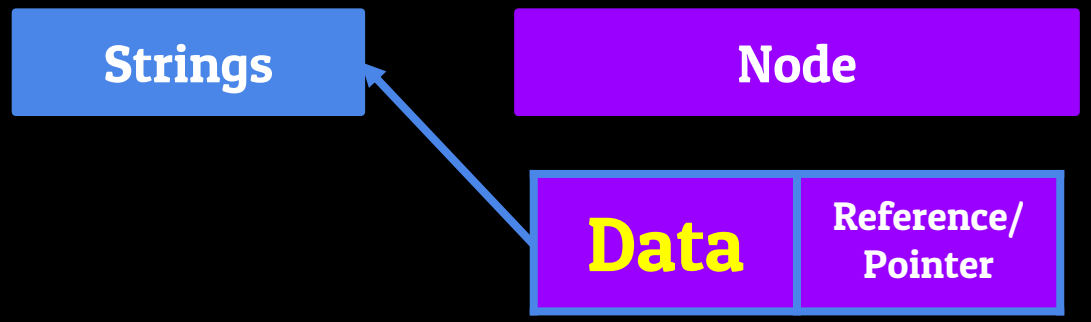
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



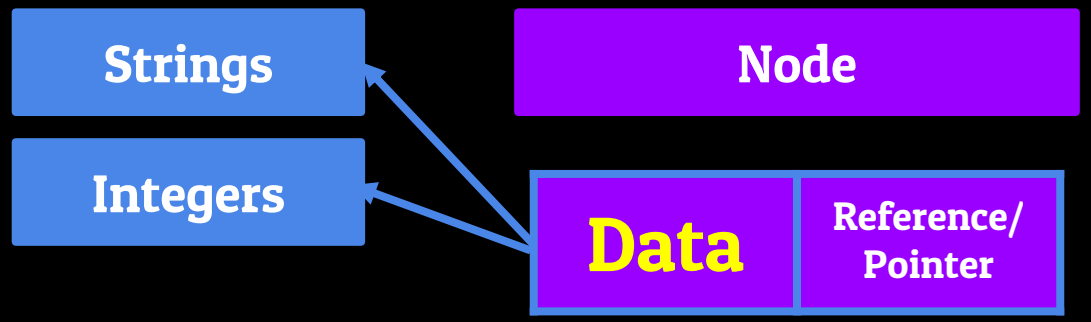
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



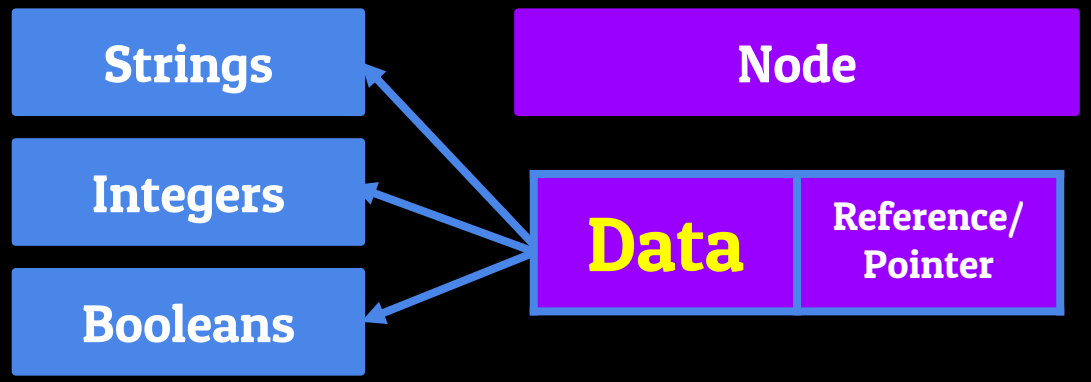
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



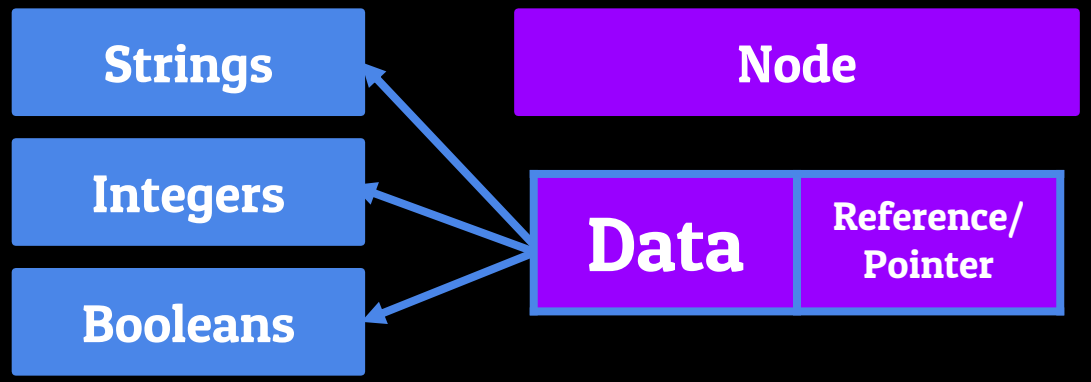
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



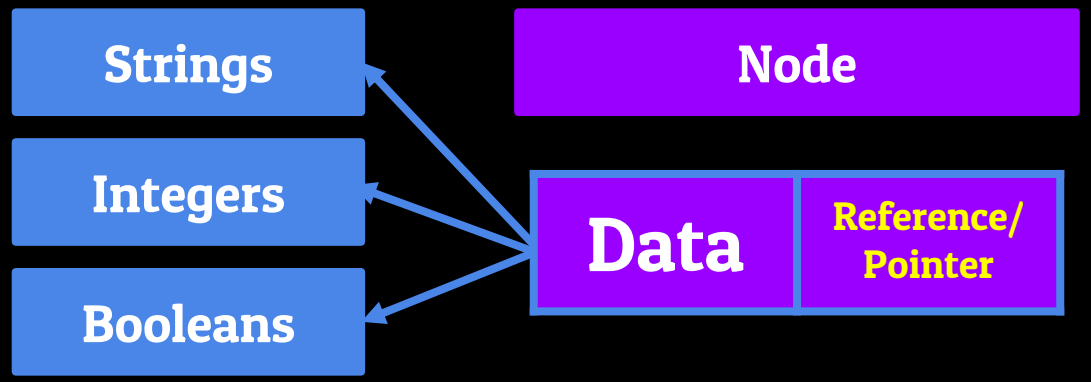
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



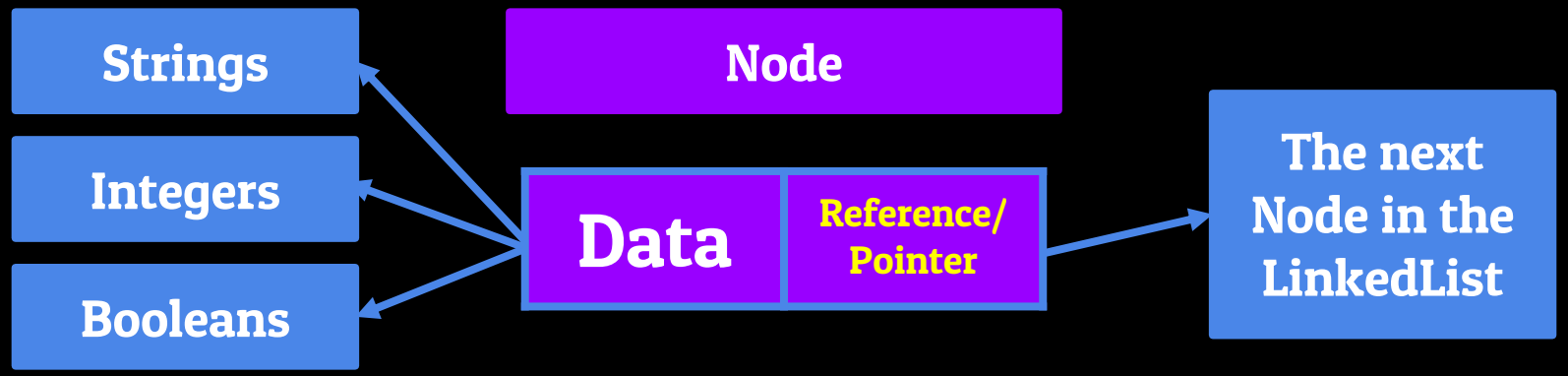
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



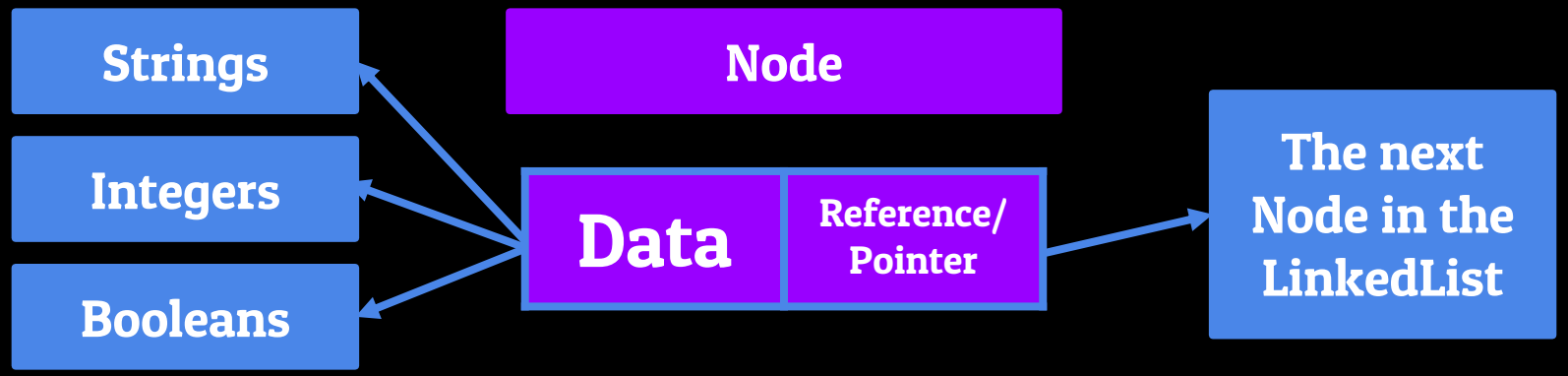
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

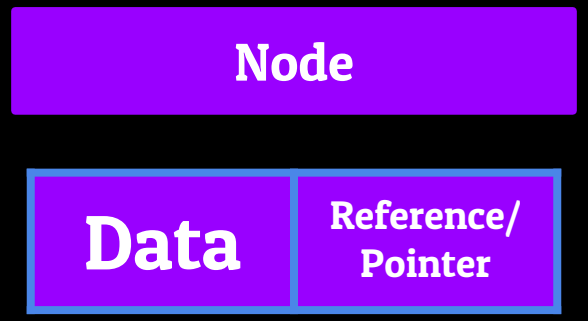
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

Node

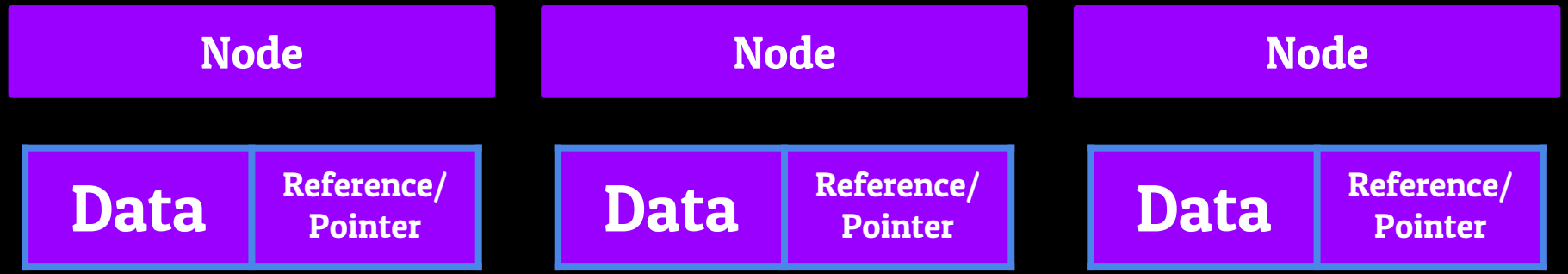
The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - LinkedList Basics

- A LinkedList is a sequential access linear data structure in which every element is a separate object called a Node, which has 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



The Linked List - Linked List Visualization

Linked List

The Linked List - Linked List Visualization

Linked List

Head Node

The Linked List - Linked List Visualization

Linked List

Head Node

Data	Reference/ Pointer

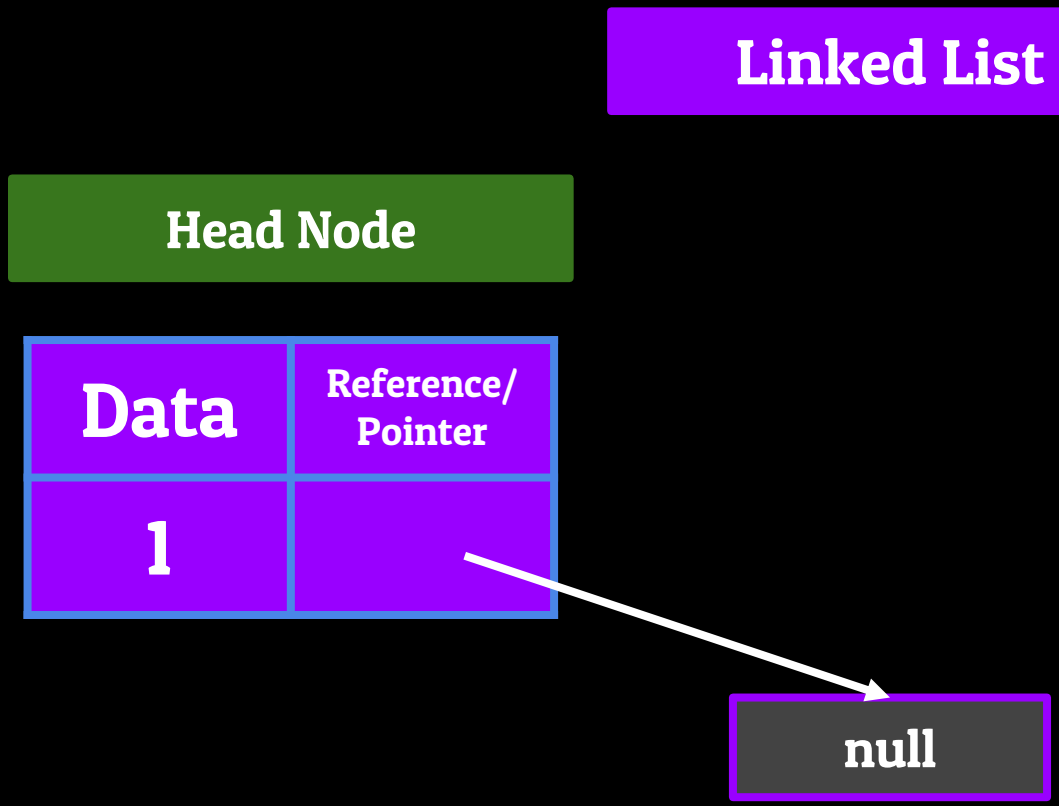
The Linked List - Linked List Visualization

Linked List

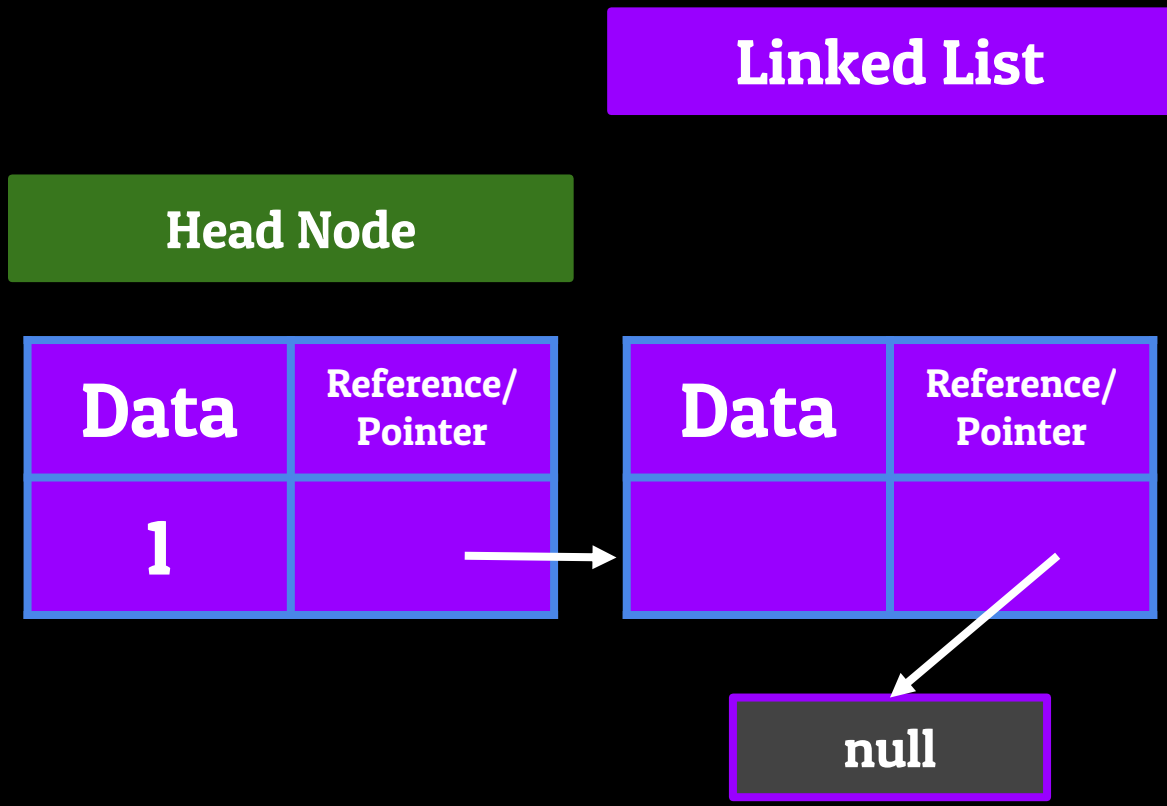
Head Node

Data	Reference/ Pointer
1	

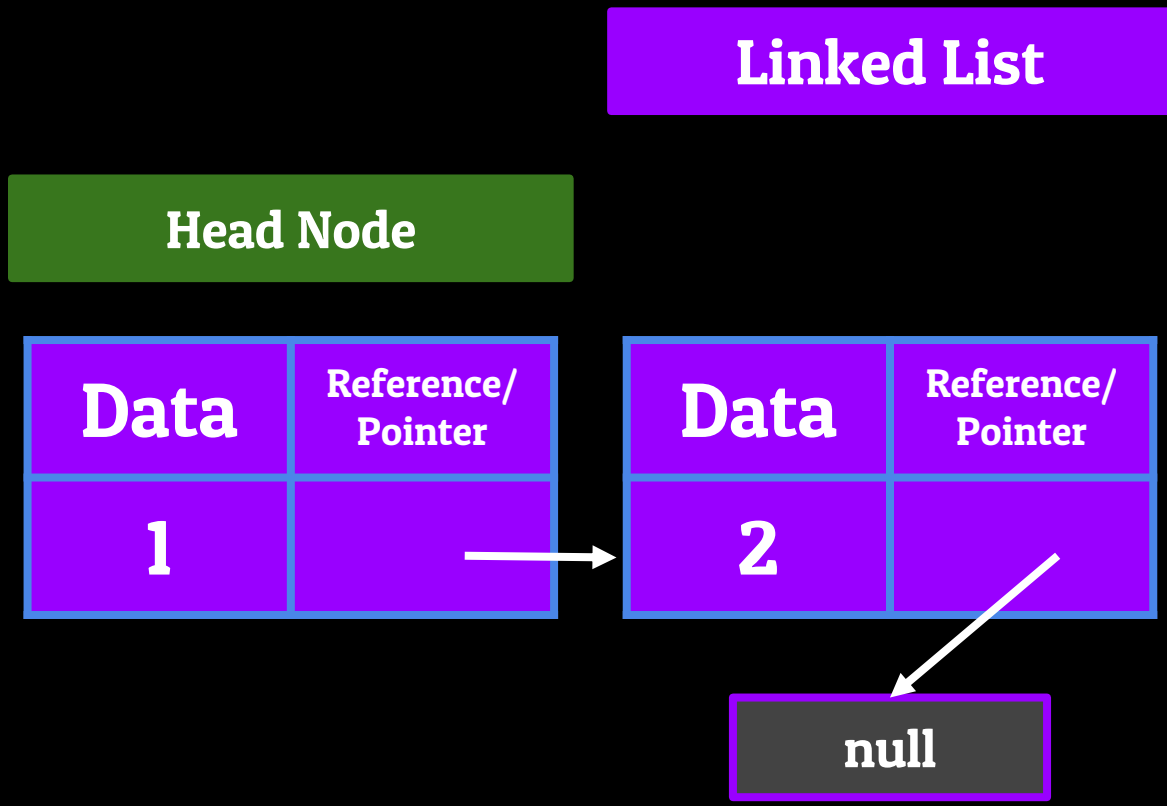
The Linked List - Linked List Visualization



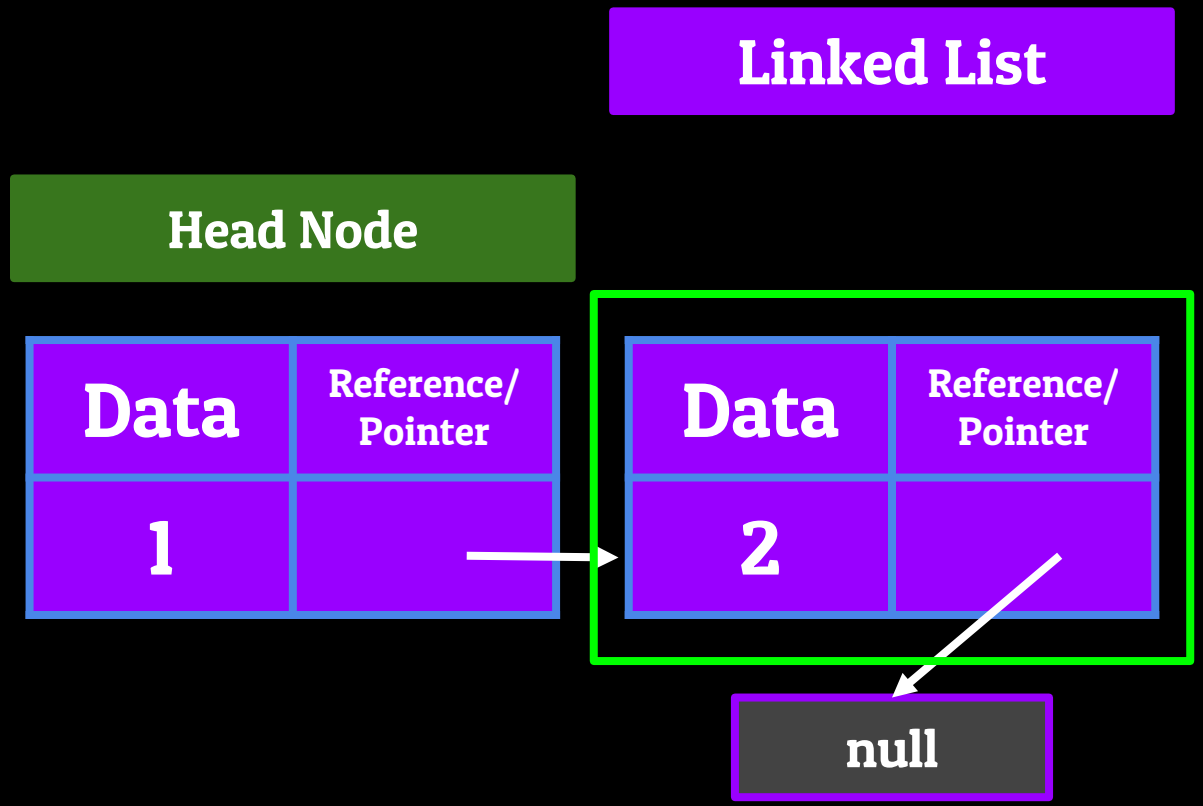
The Linked List - Linked List Visualization



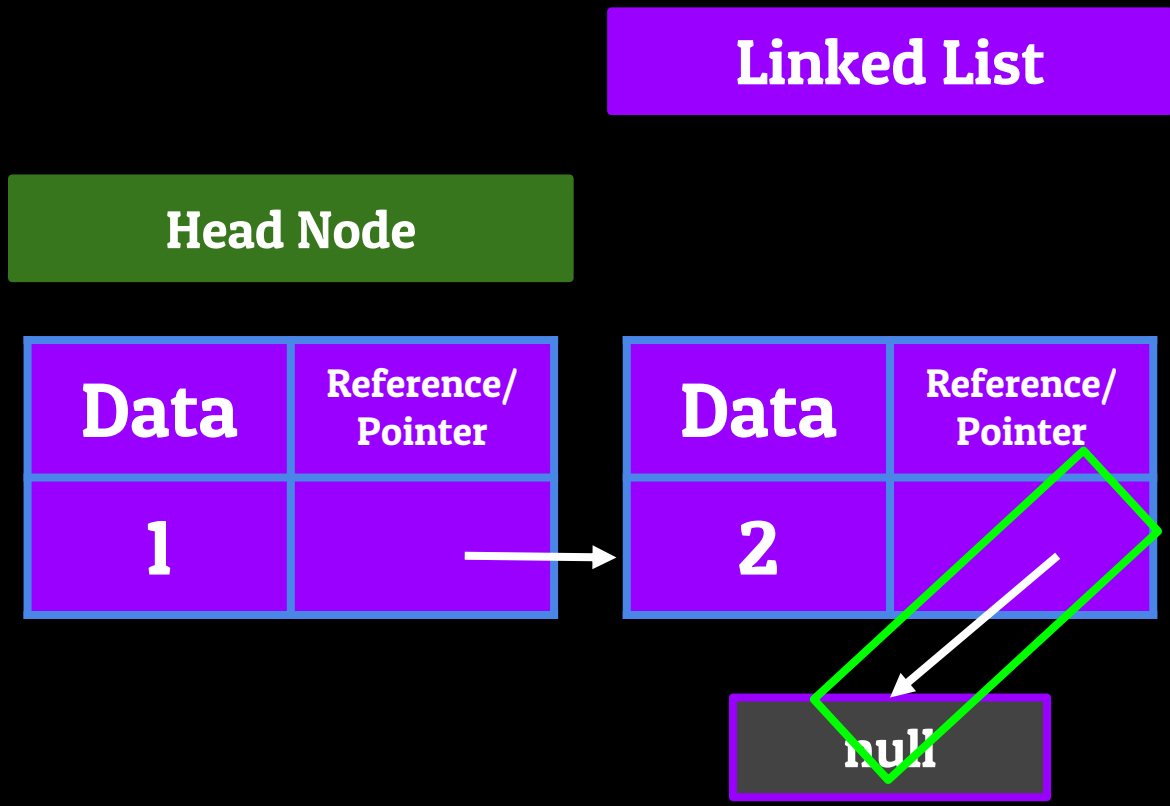
The Linked List - Linked List Visualization



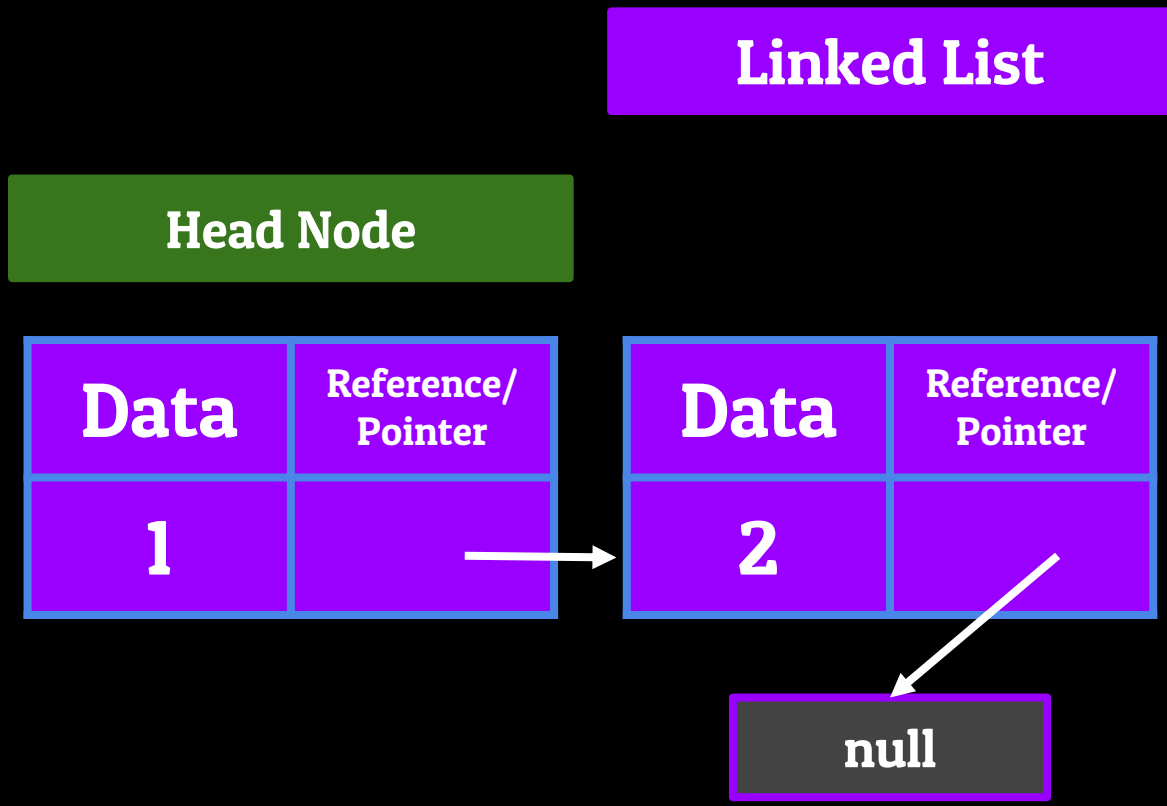
The Linked List - Linked List Visualization



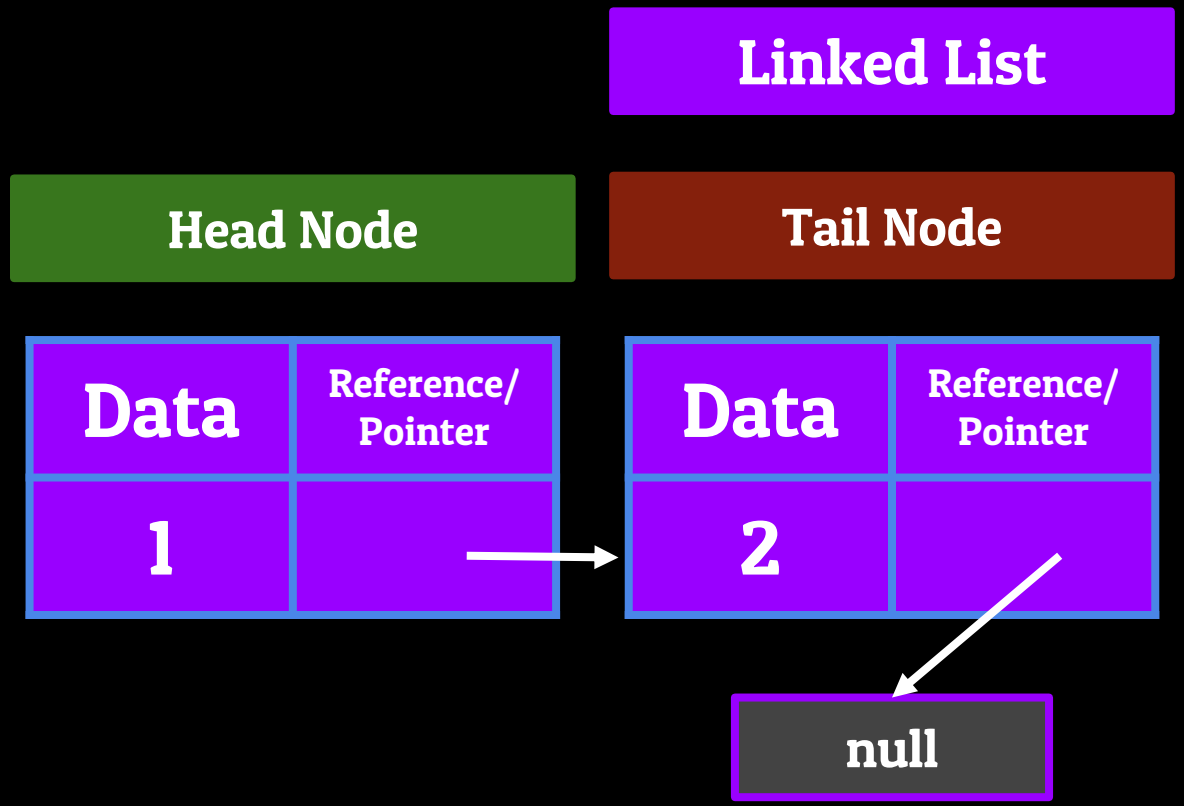
The Linked List - Linked List Visualization



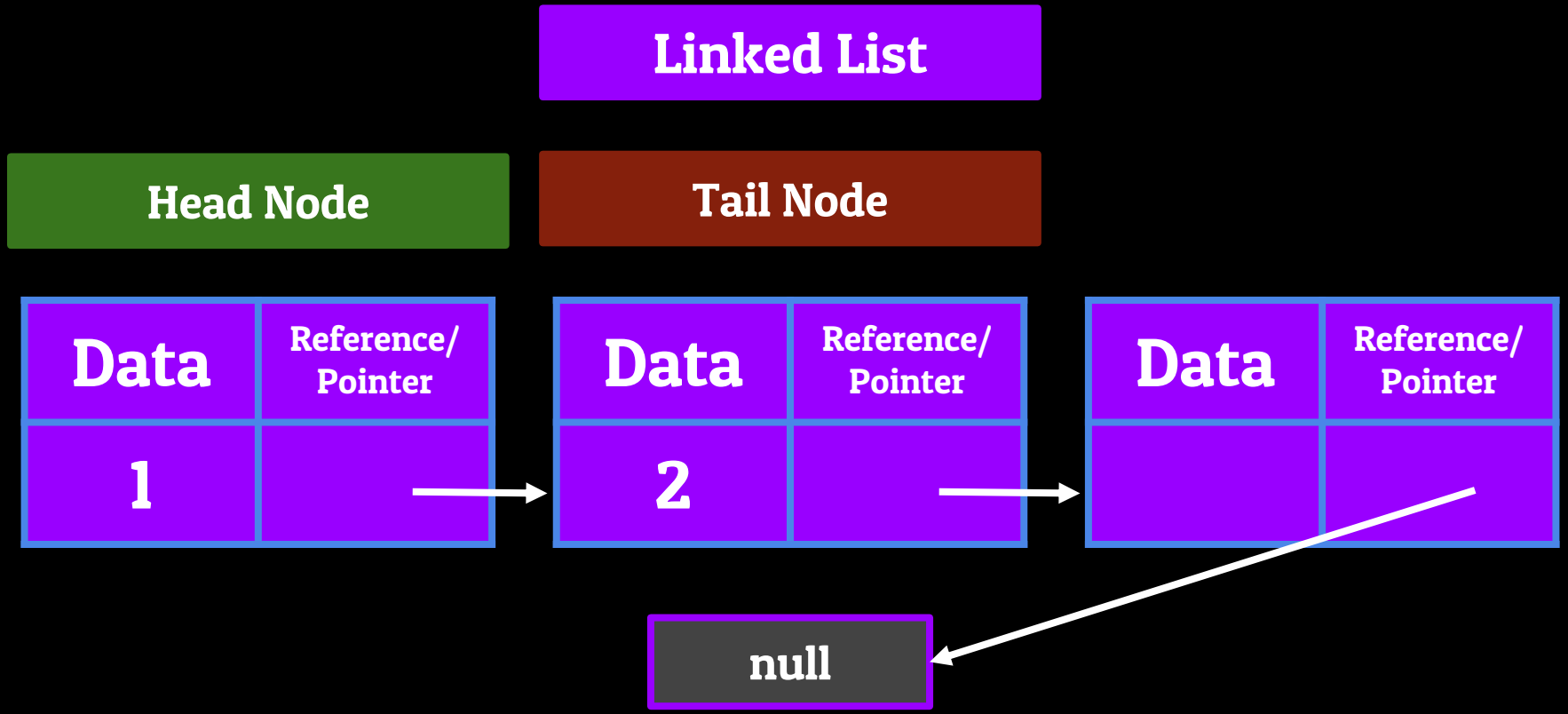
The Linked List - Linked List Visualization



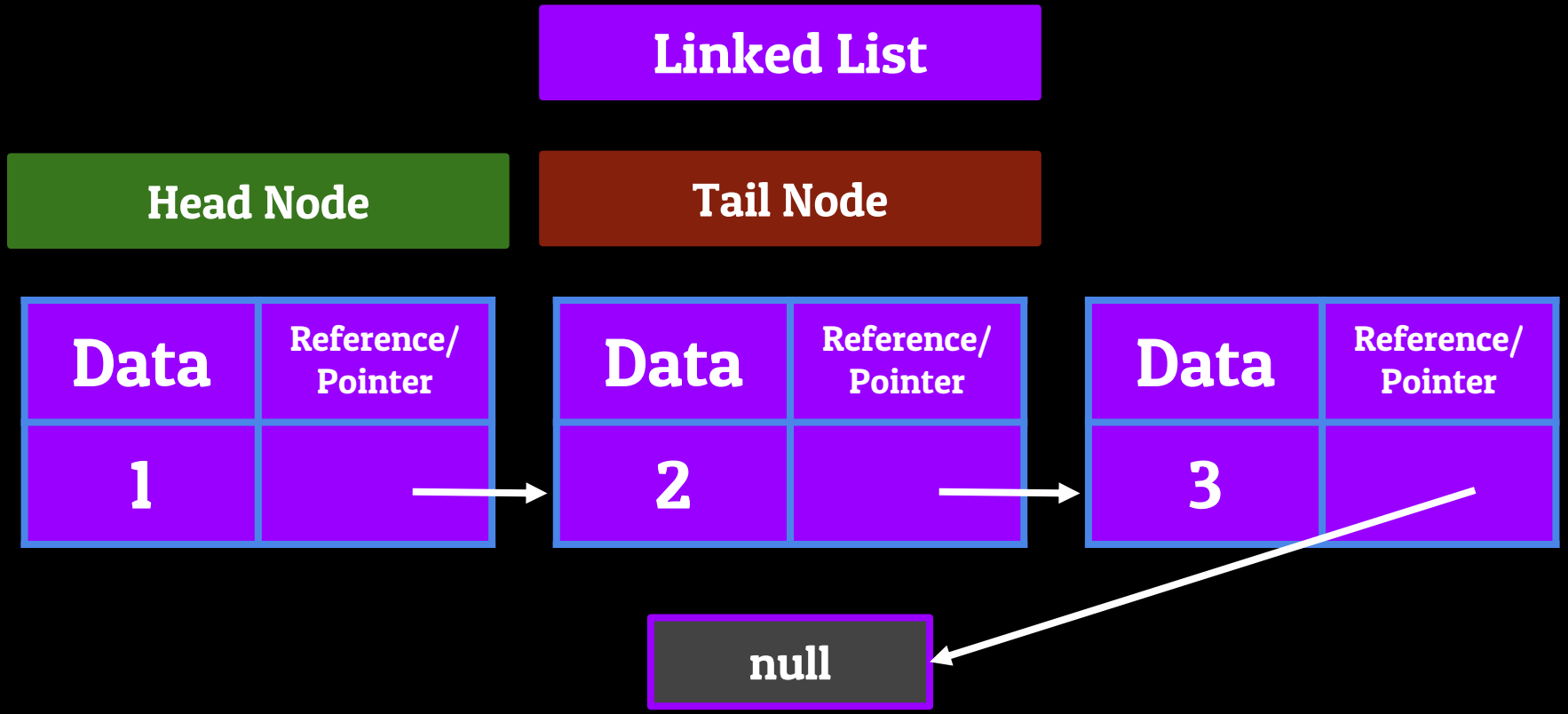
The Linked List - Linked List Visualization



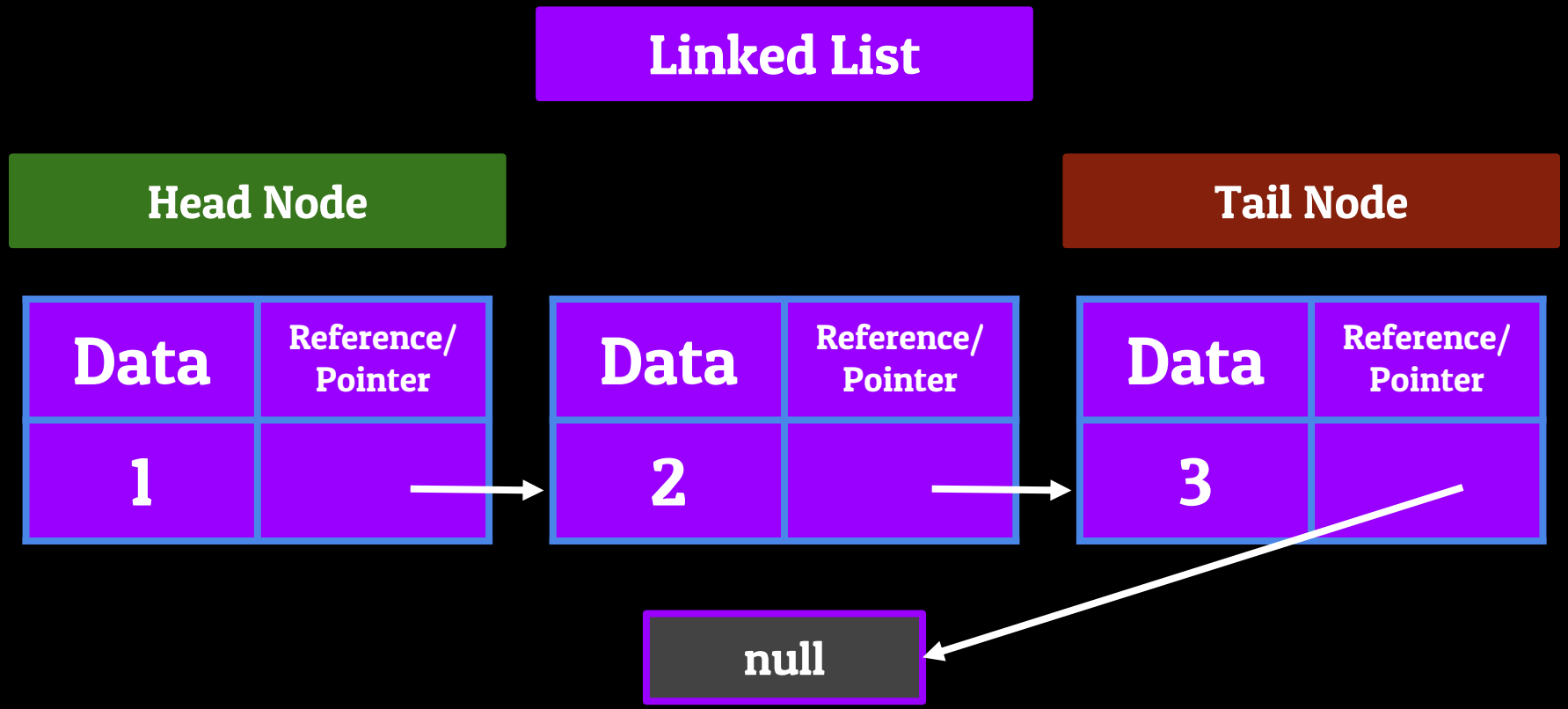
The Linked List - Linked List Visualization



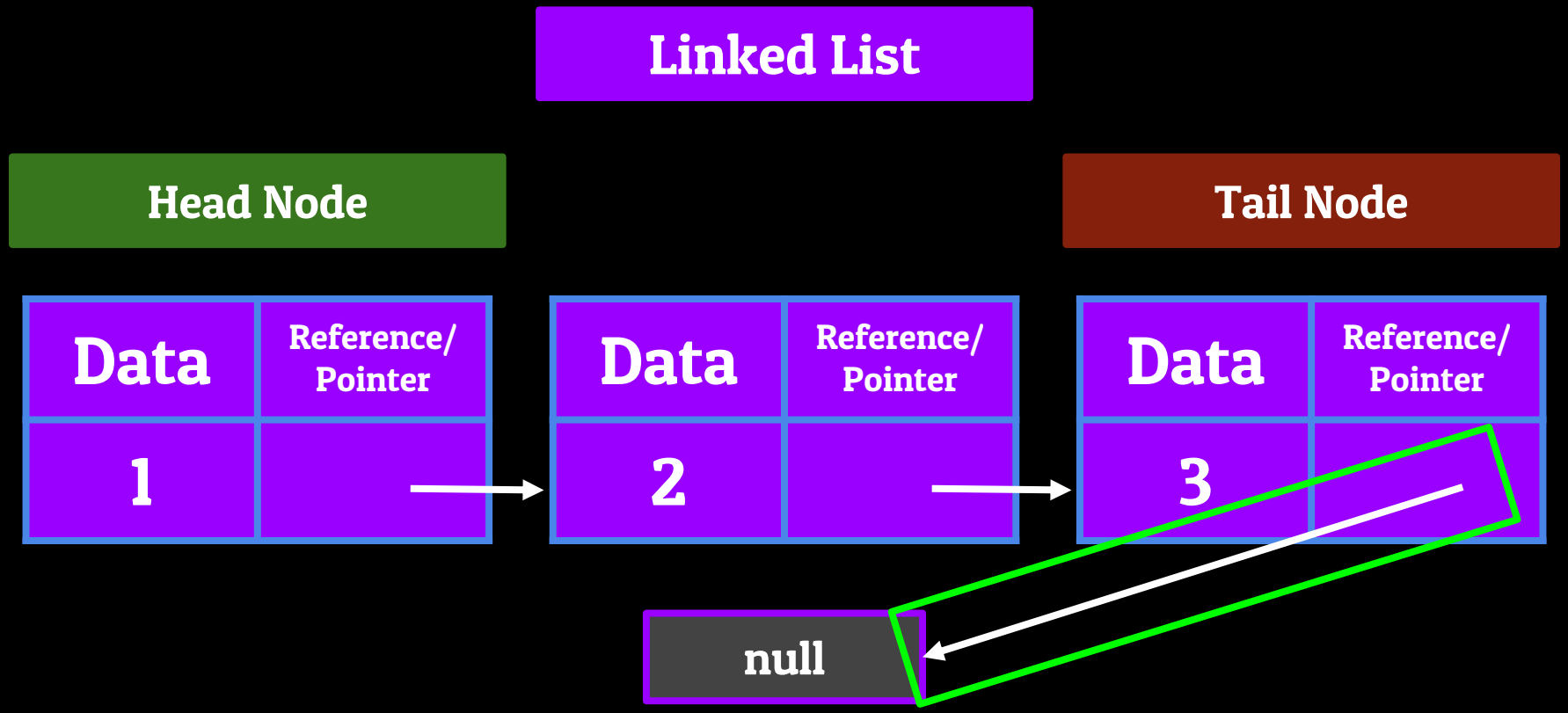
The Linked List - Linked List Visualization



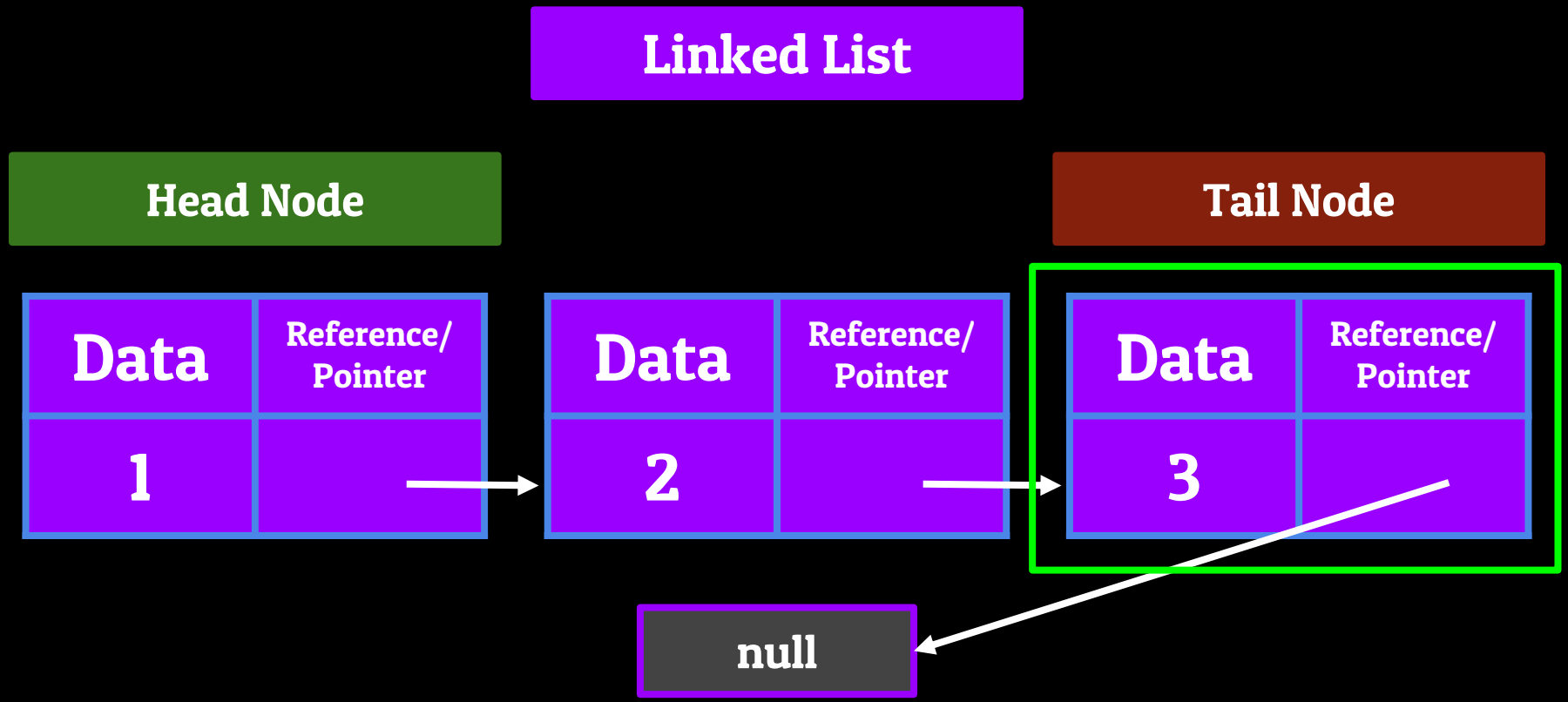
The Linked List - Linked List Visualization



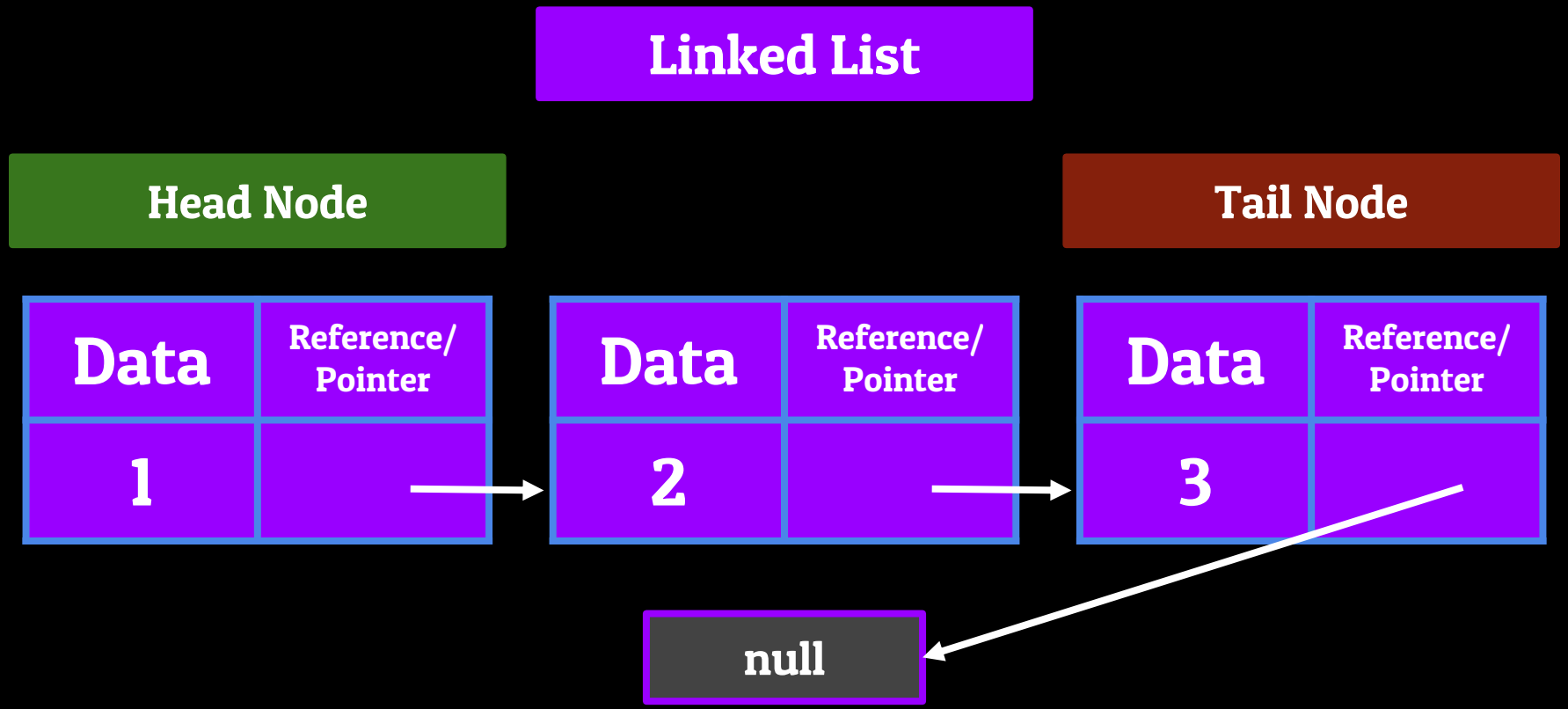
The Linked List - Linked List Visualization



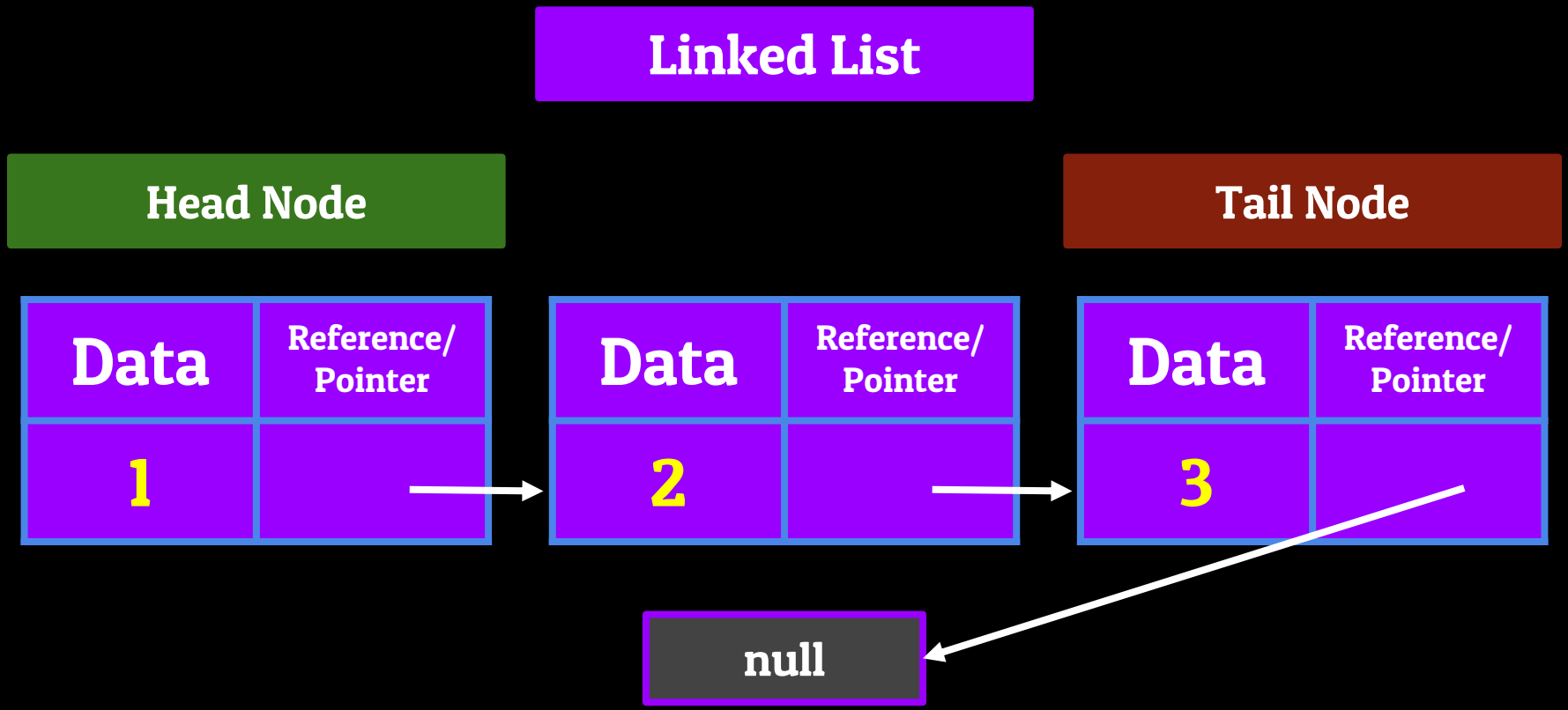
The Linked List - Linked List Visualization



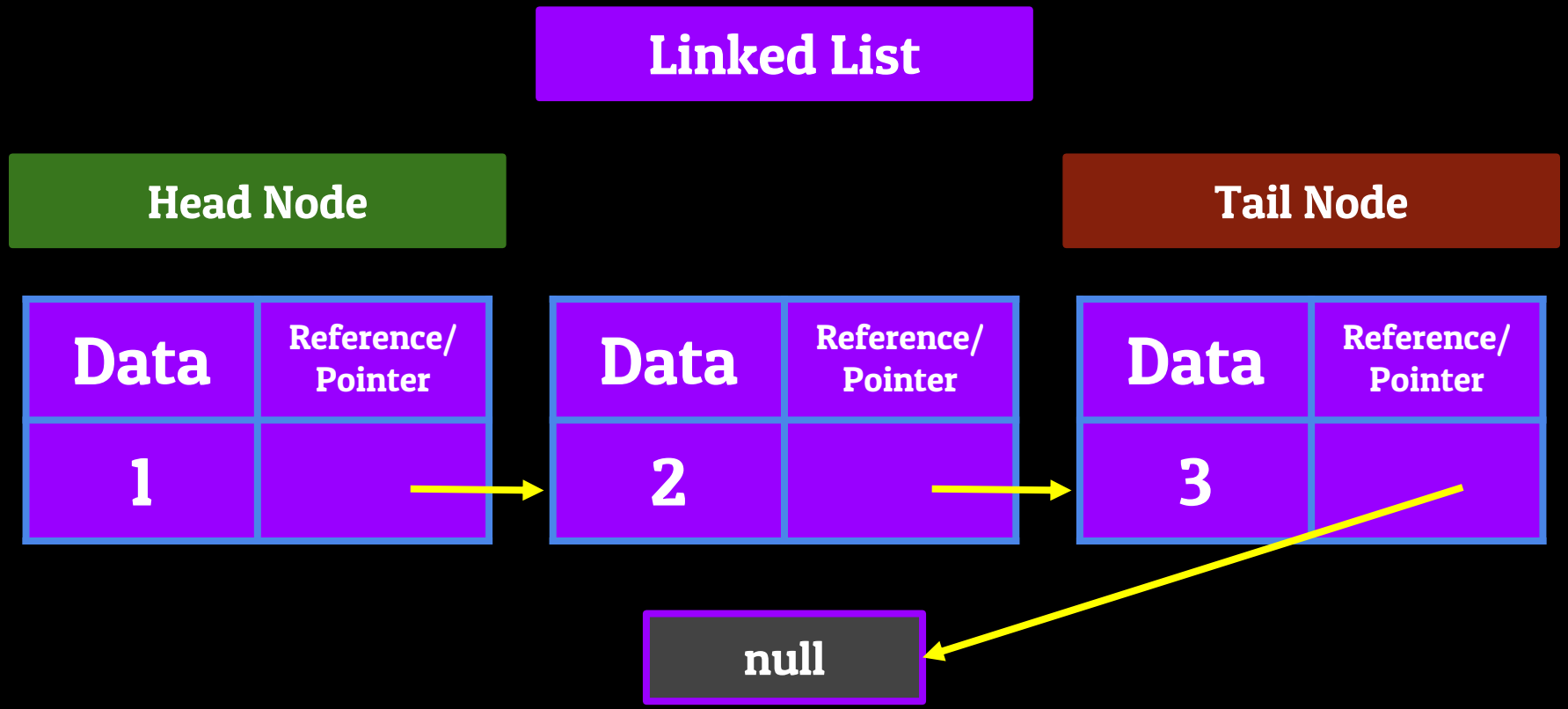
The Linked List - Linked List Visualization



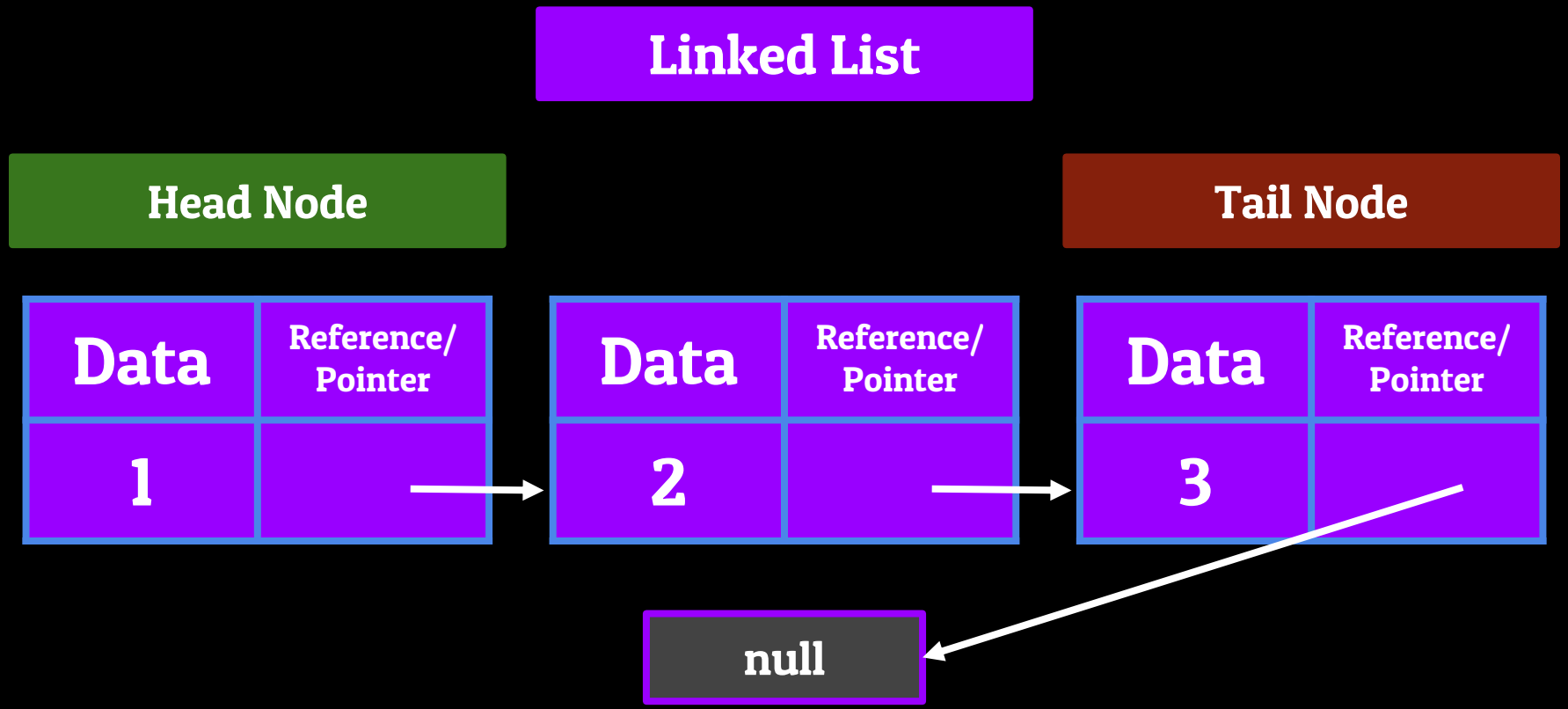
The Linked List - Linked List Visualization



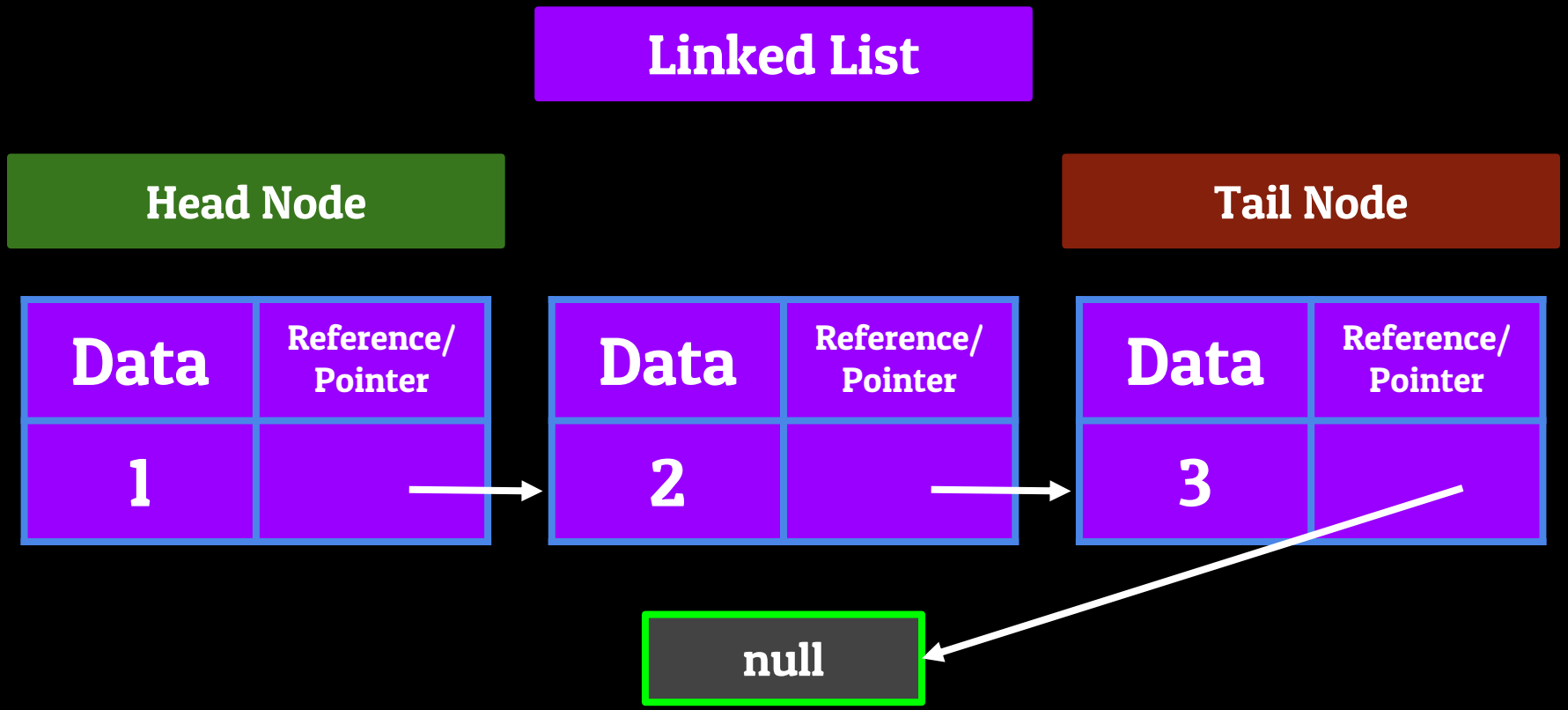
The Linked List - Linked List Visualization



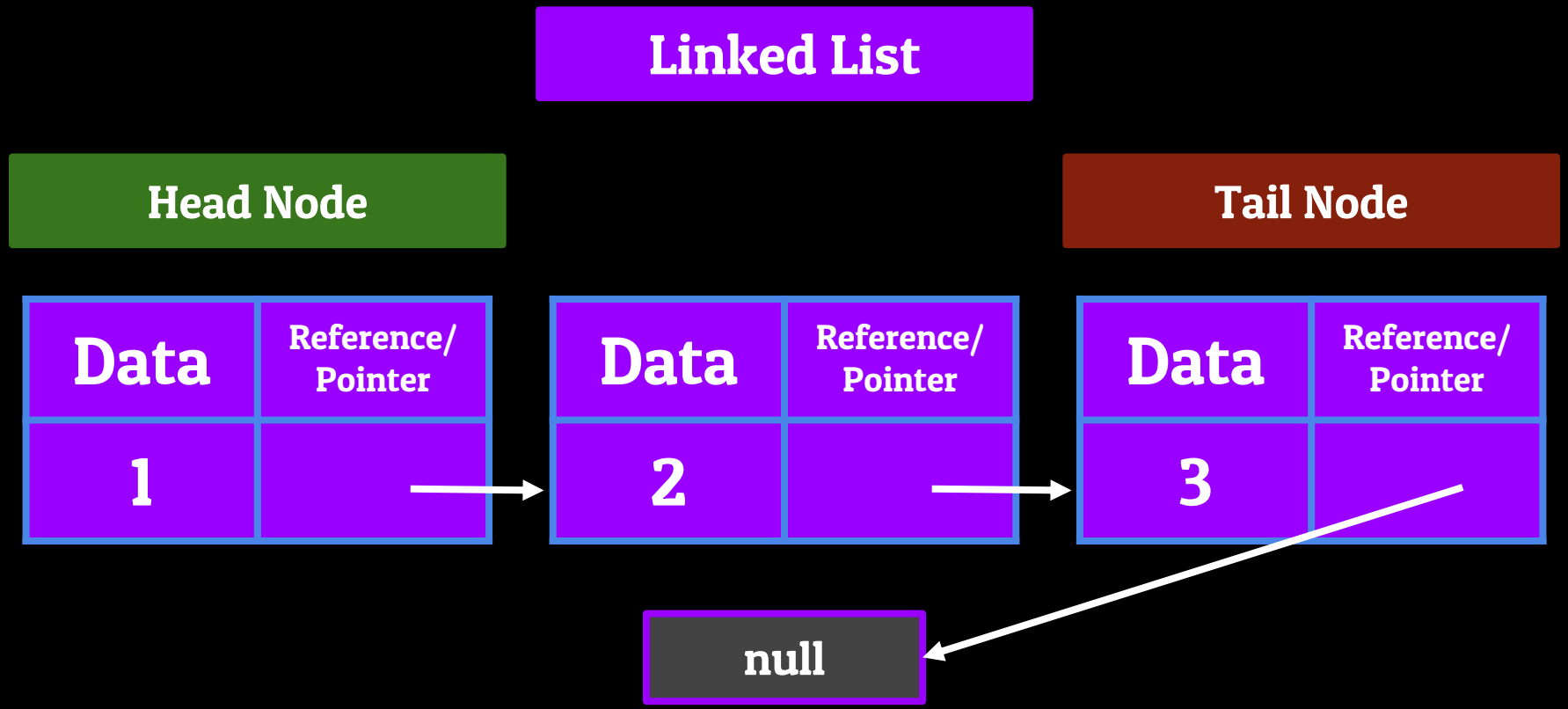
The Linked List - Linked List Visualization



The Linked List - Linked List Visualization

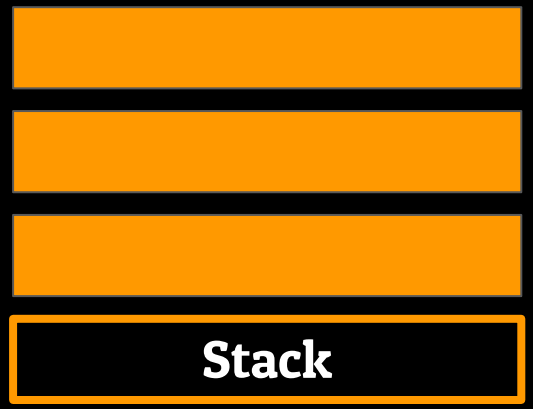


The Linked List - Linked List Visualization

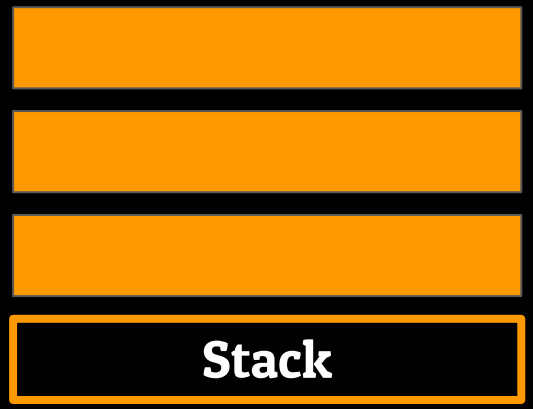


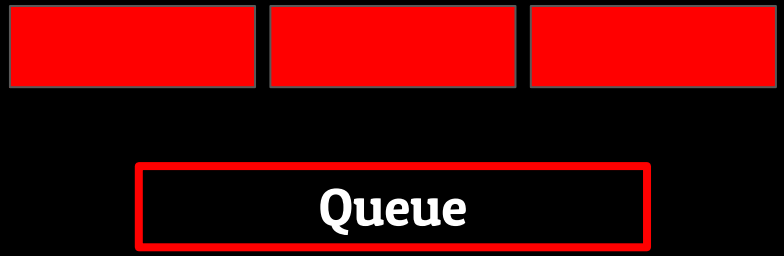
The Linked List - Adding and Removing Information

The Linked List - Adding and Removing Information



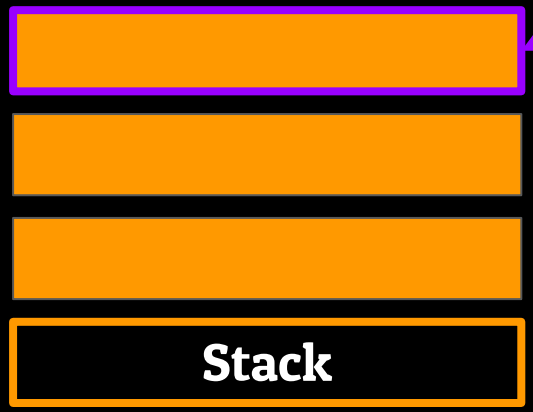
The Linked List - Adding and Removing Information

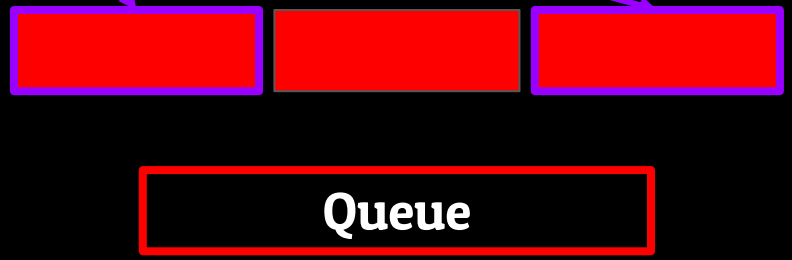




The Linked List - Adding and Removing Information

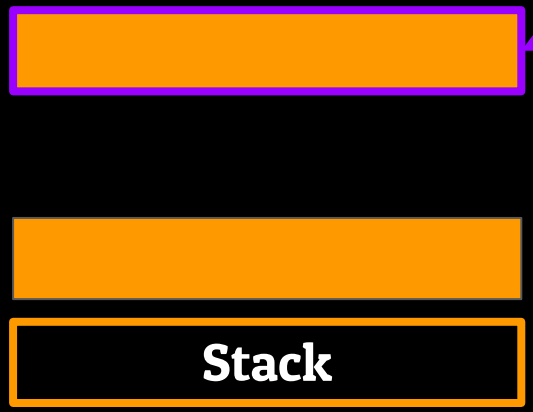
Data can only flow in and out of a few specified points

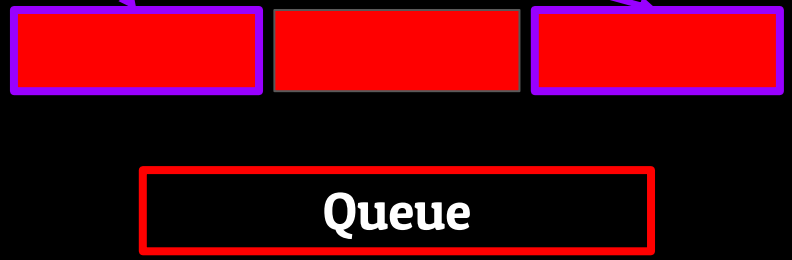




The Linked List - Adding and Removing Information

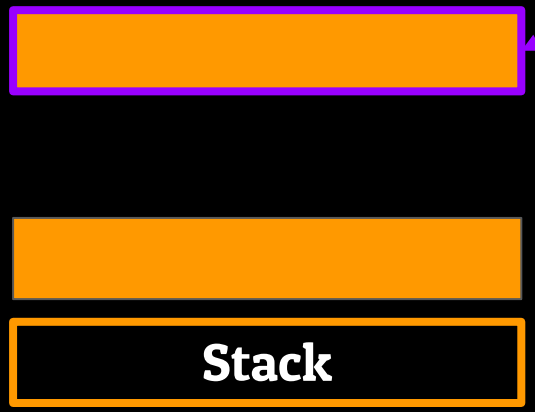
Data can only flow in and out of a few specified points

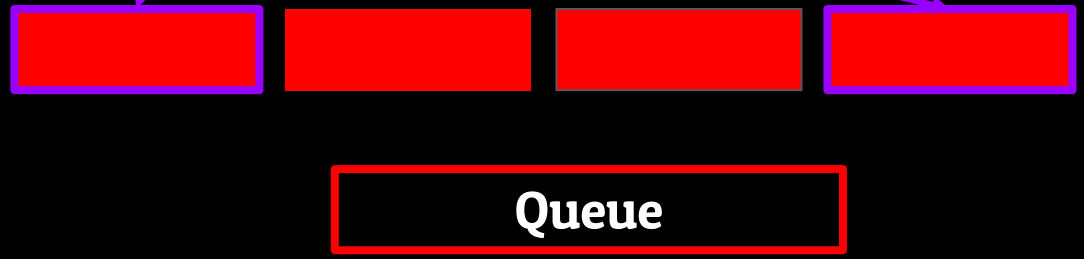




The Linked List - Adding and Removing Information

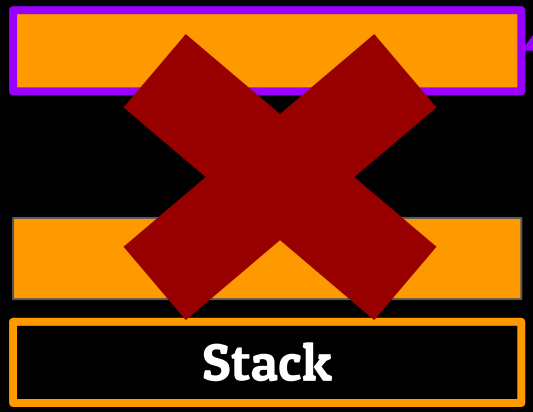
Data can only flow in and out of a few specified points

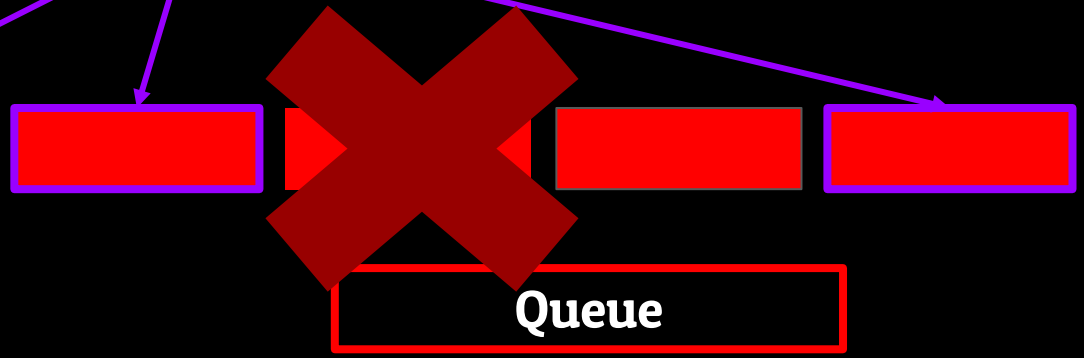




The Linked List - Adding and Removing Information

Data can only flow in and out of a few specified points





The Linked List - Adding and Removing Information

The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

Add to Head

Remove from Head

The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

Add to Head

Add to the Middle

Remove from Head

Remove from the Middle

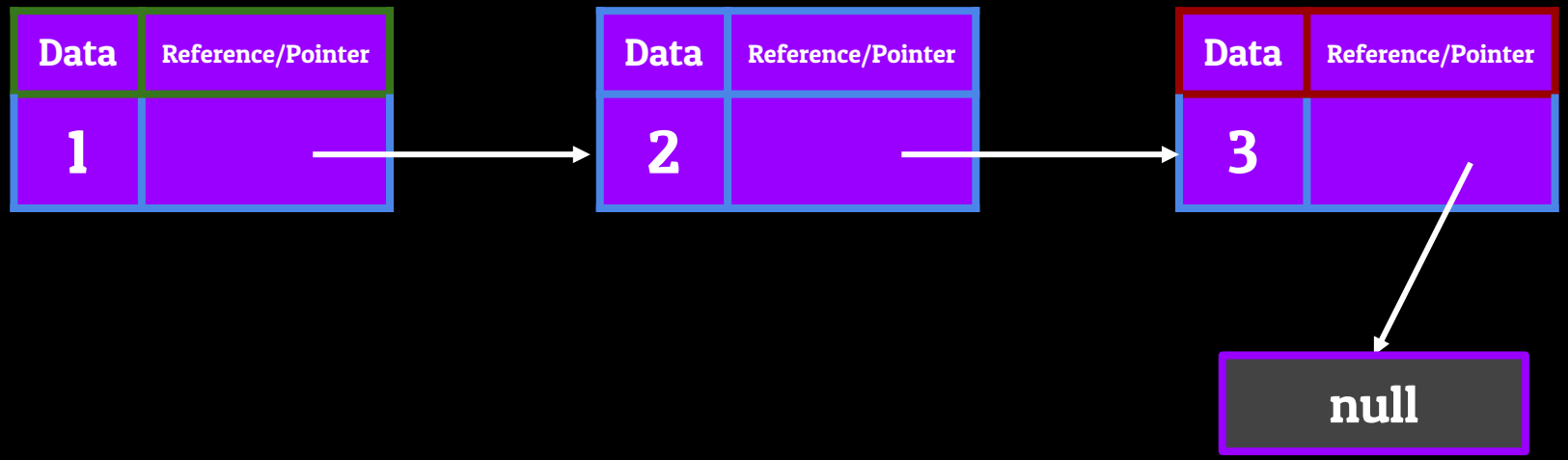
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

Add to Head	Add to the Middle	Add to Tail
Remove from Head	Remove from the Middle	Remove from Tail

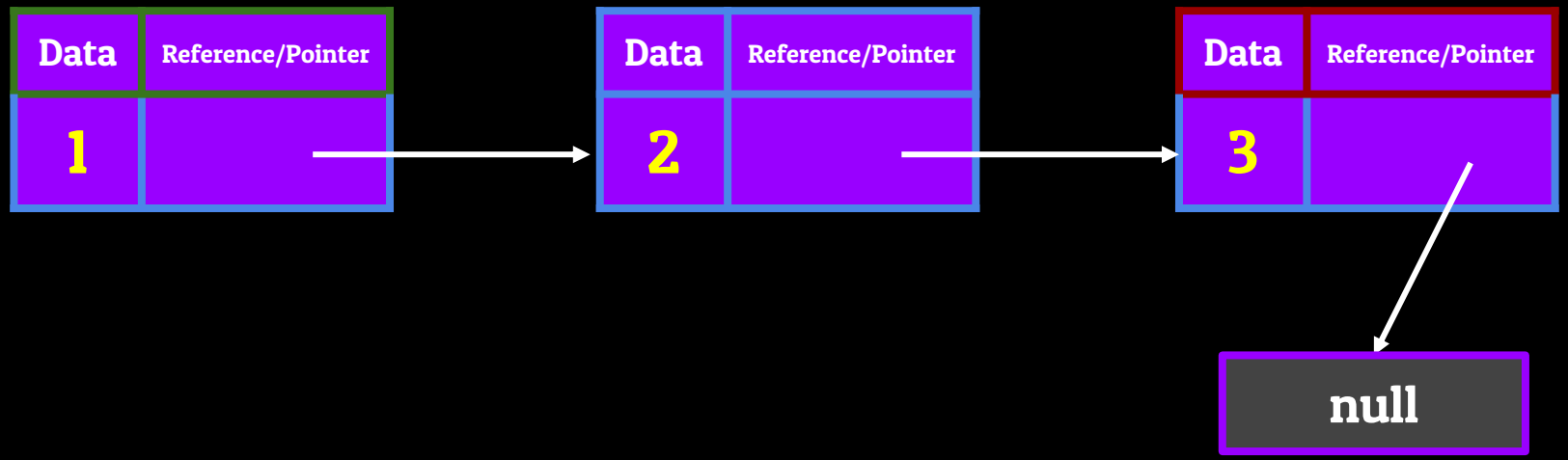
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



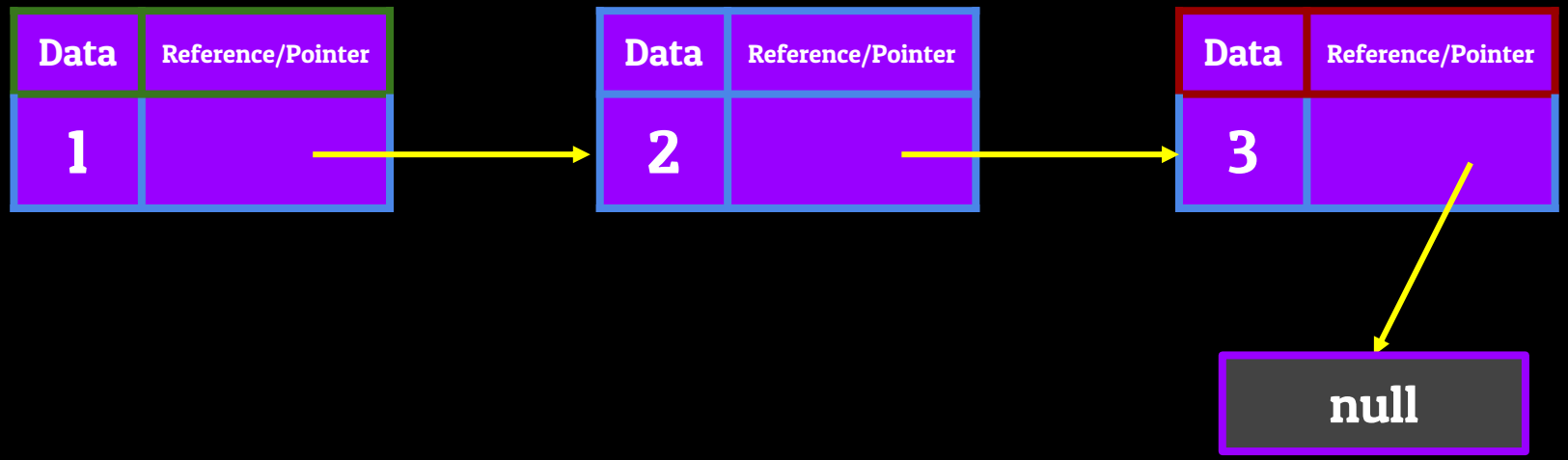
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



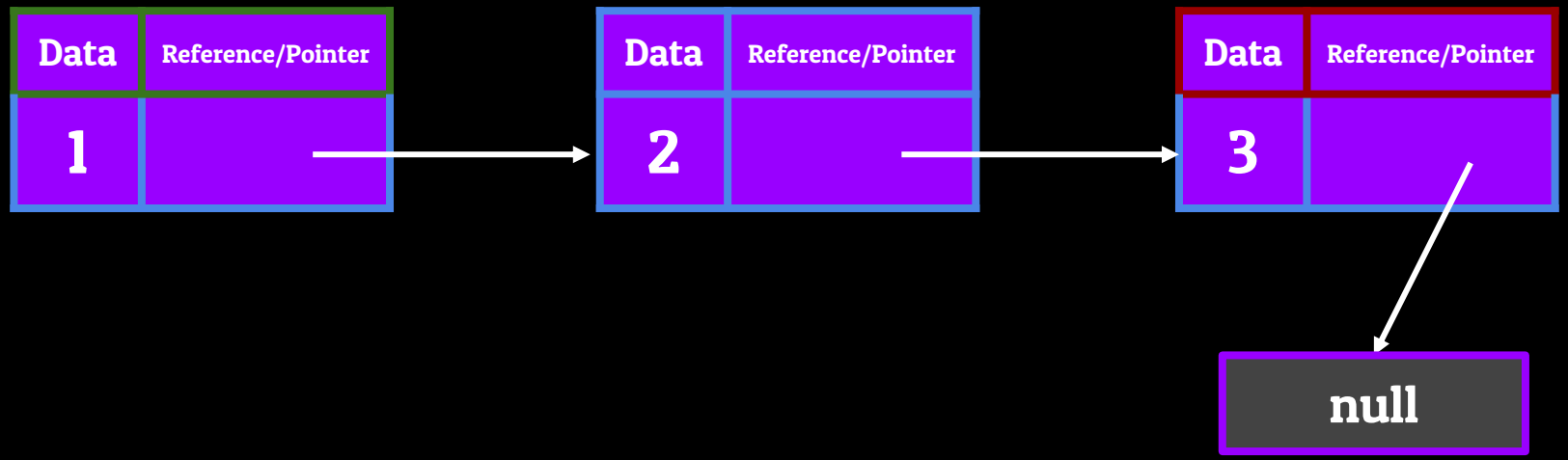
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



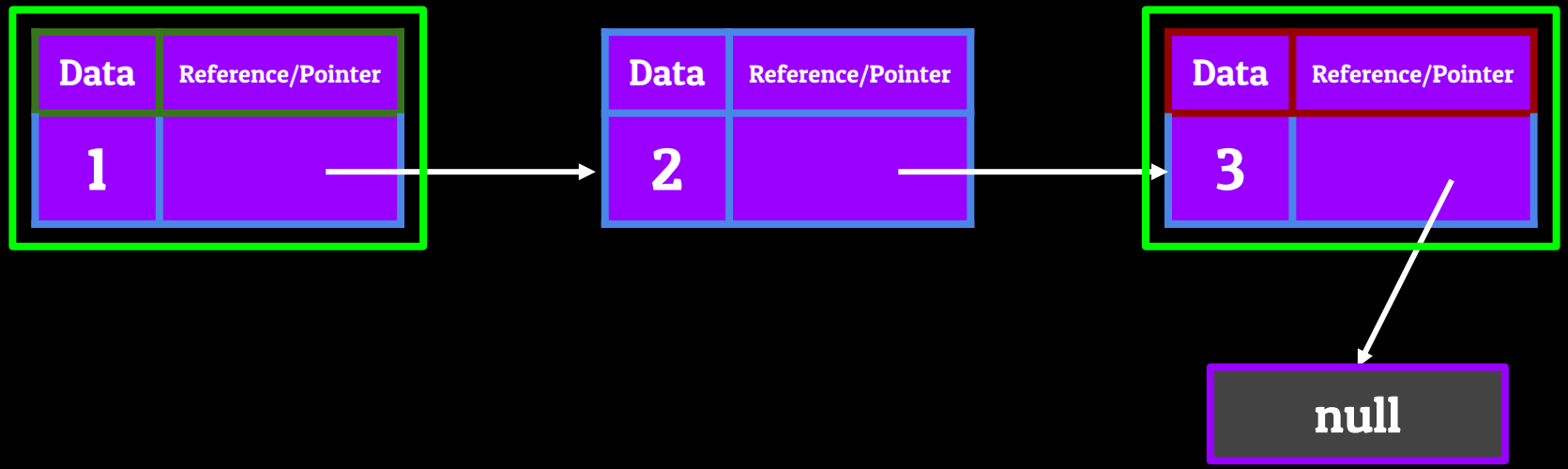
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



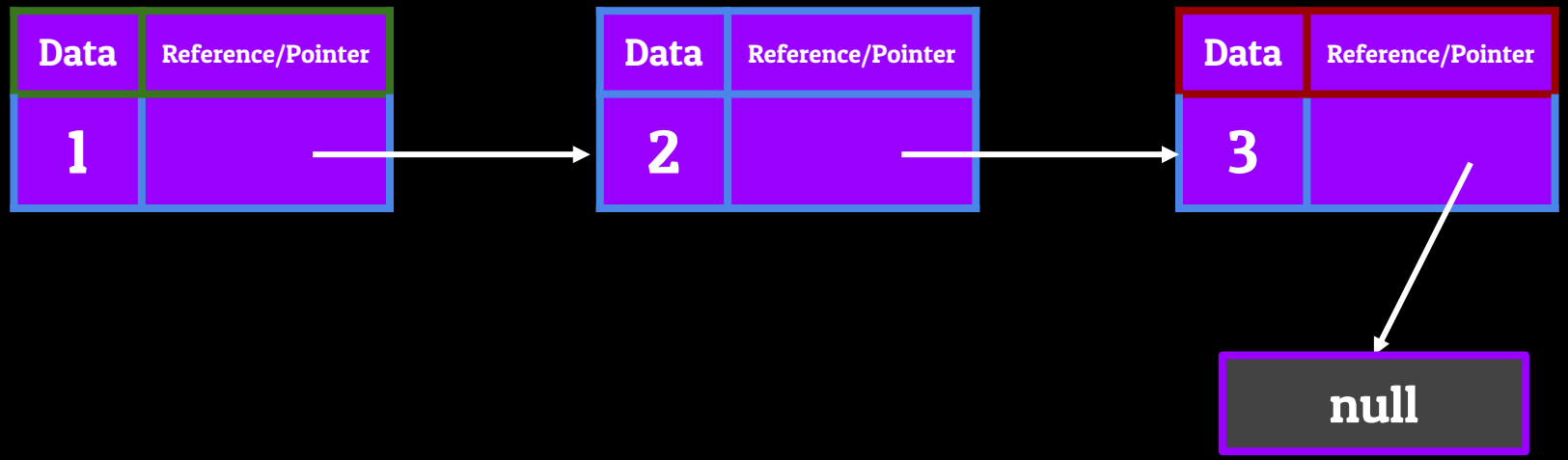
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



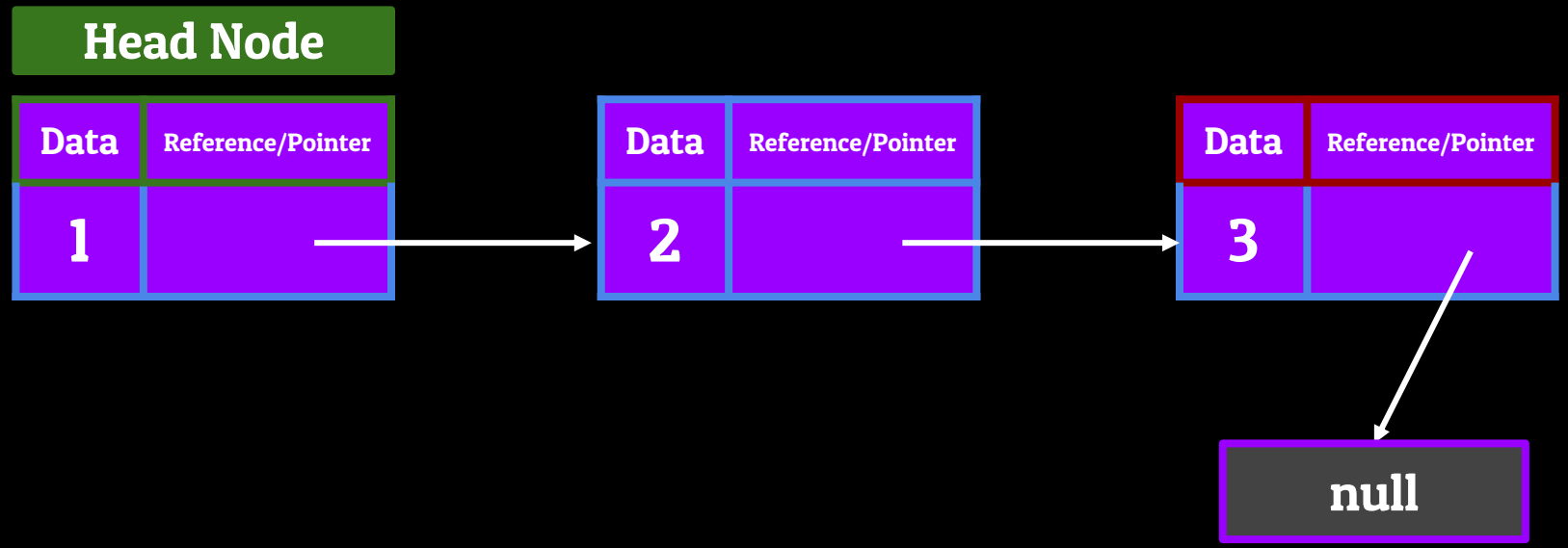
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



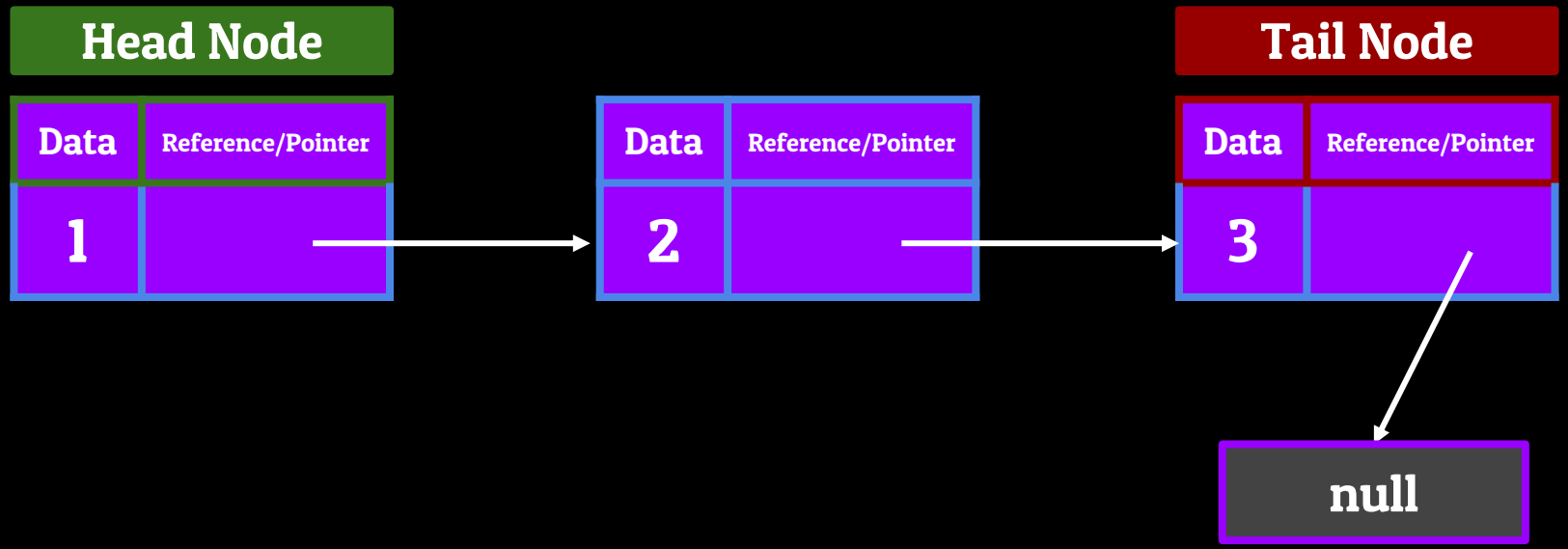
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



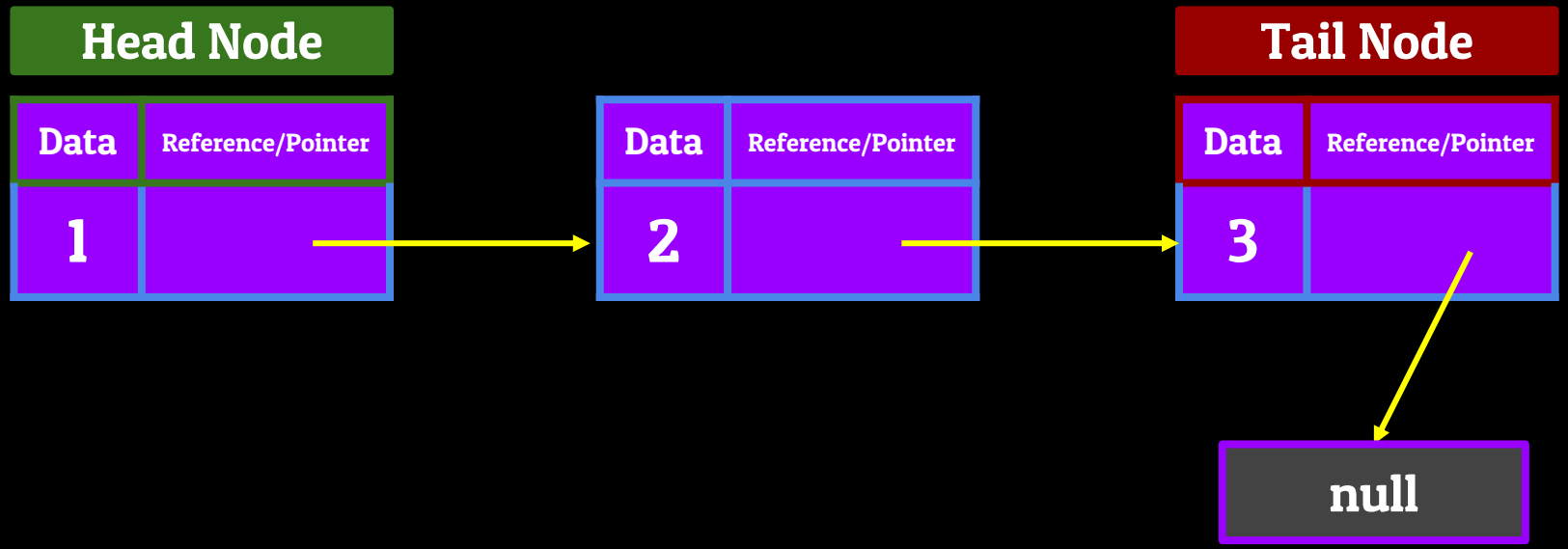
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList



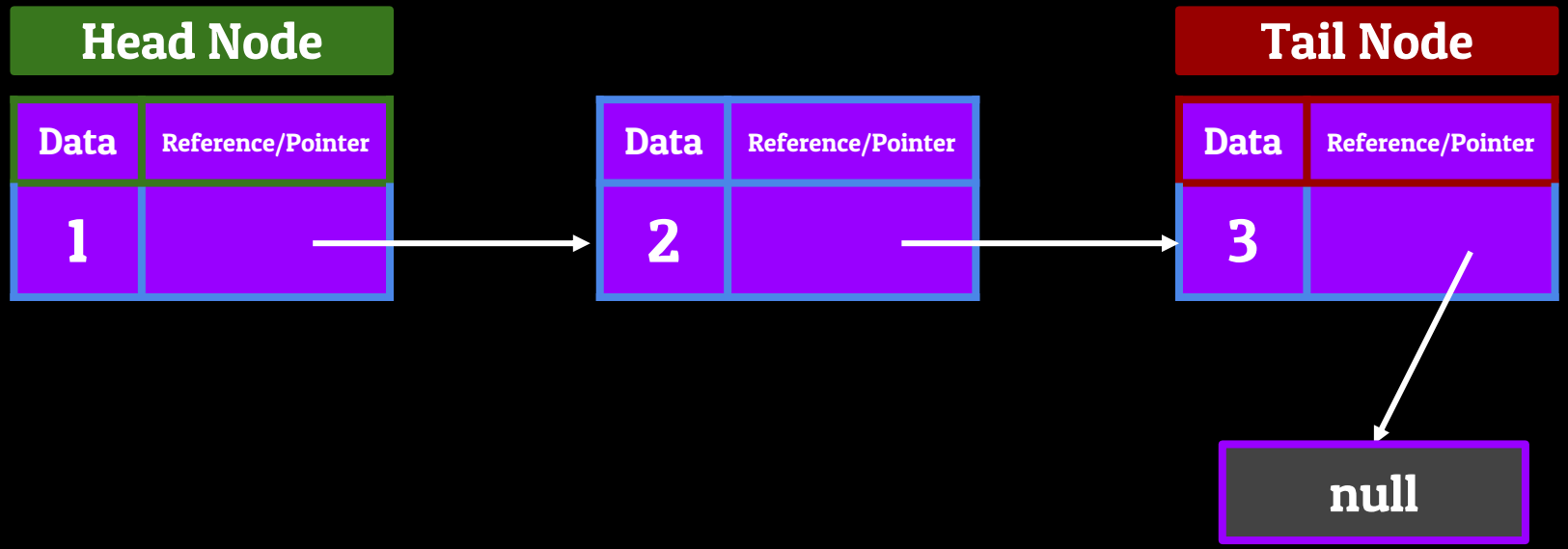
The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

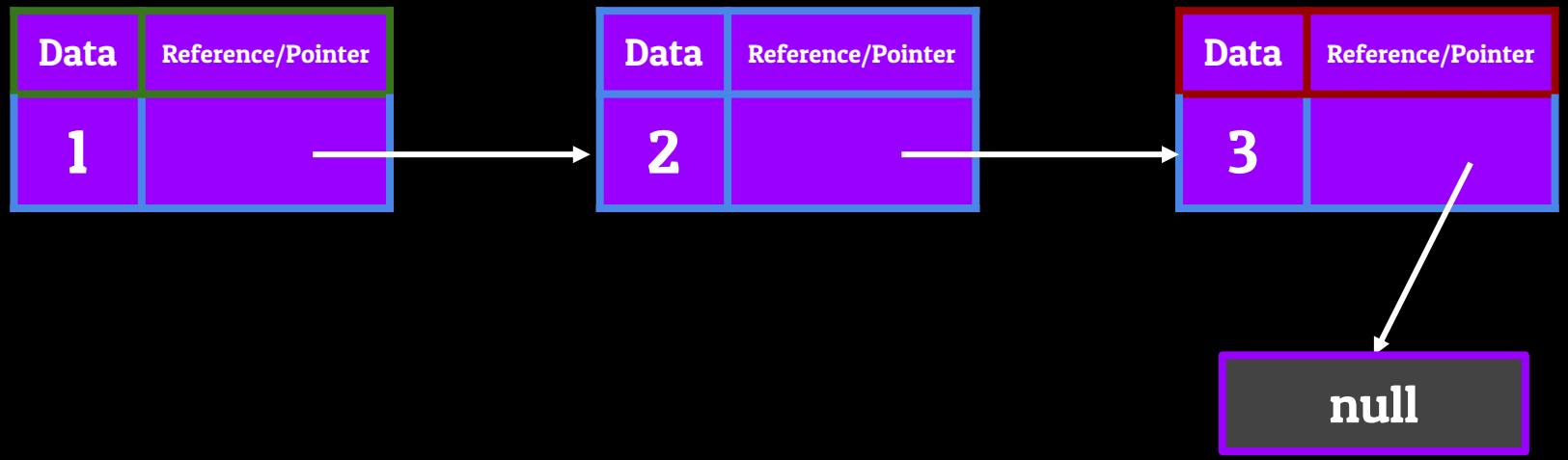


The Linked List - Adding and Removing Information

Data can flow in and out of any point of a LinkedList

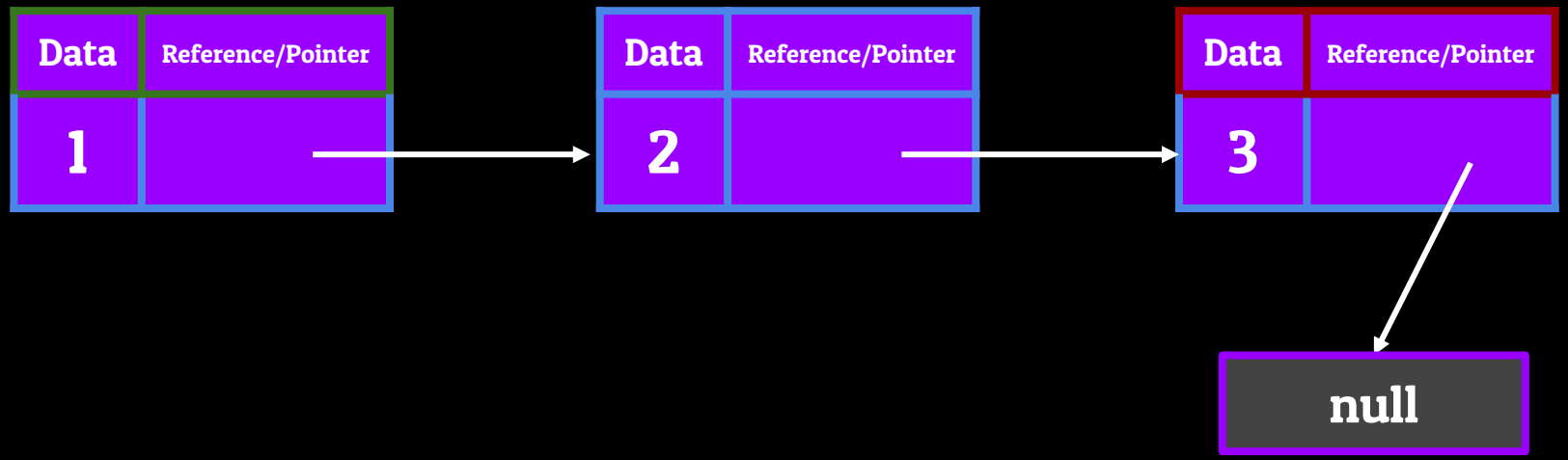


The Linked List - Adding and Removing Information



The Linked List - Adding and Removing Information

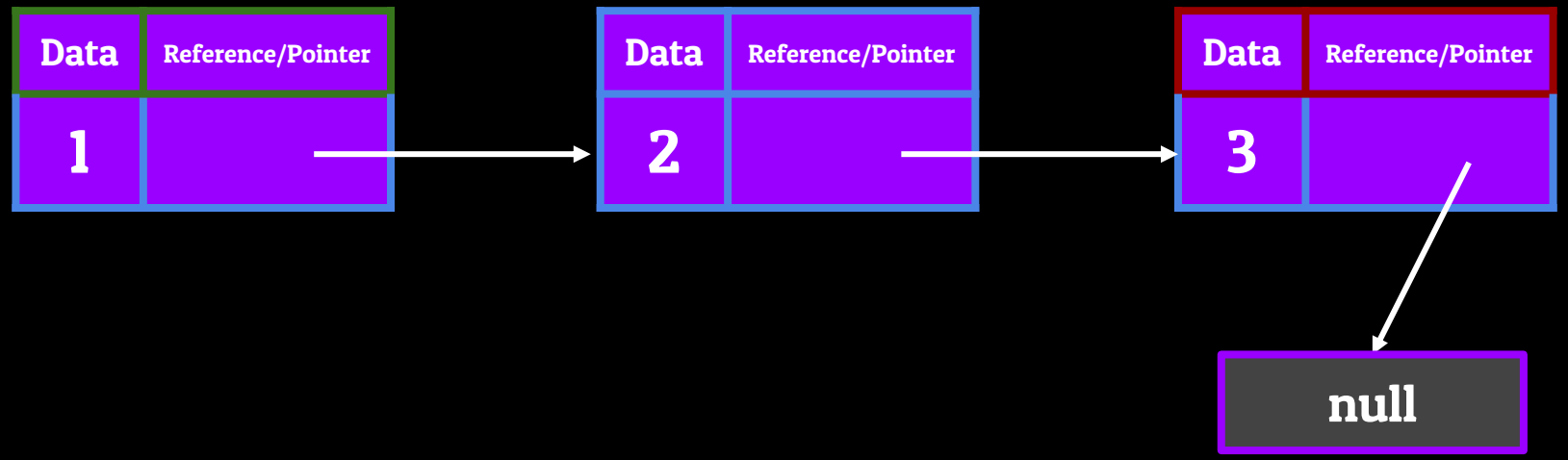
Adding to the Head of a LinkedList



The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

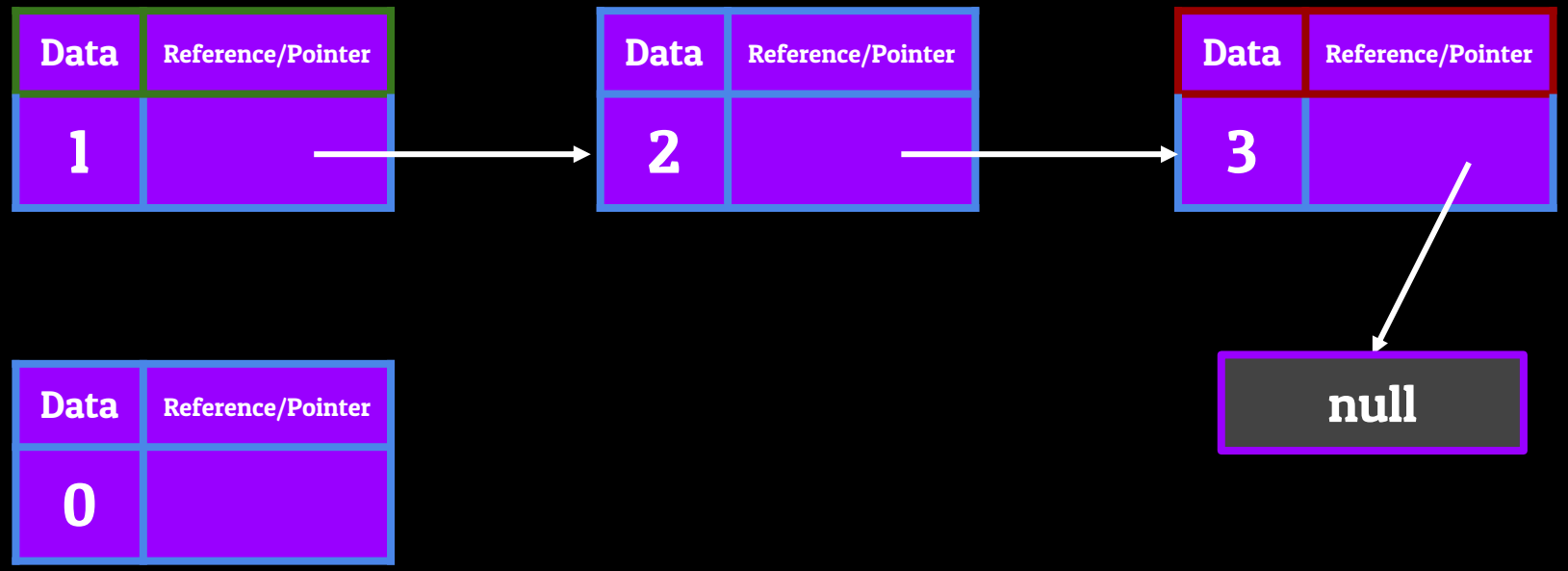
Make that new Node's pointer point to the current Head of the LinkedList



The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

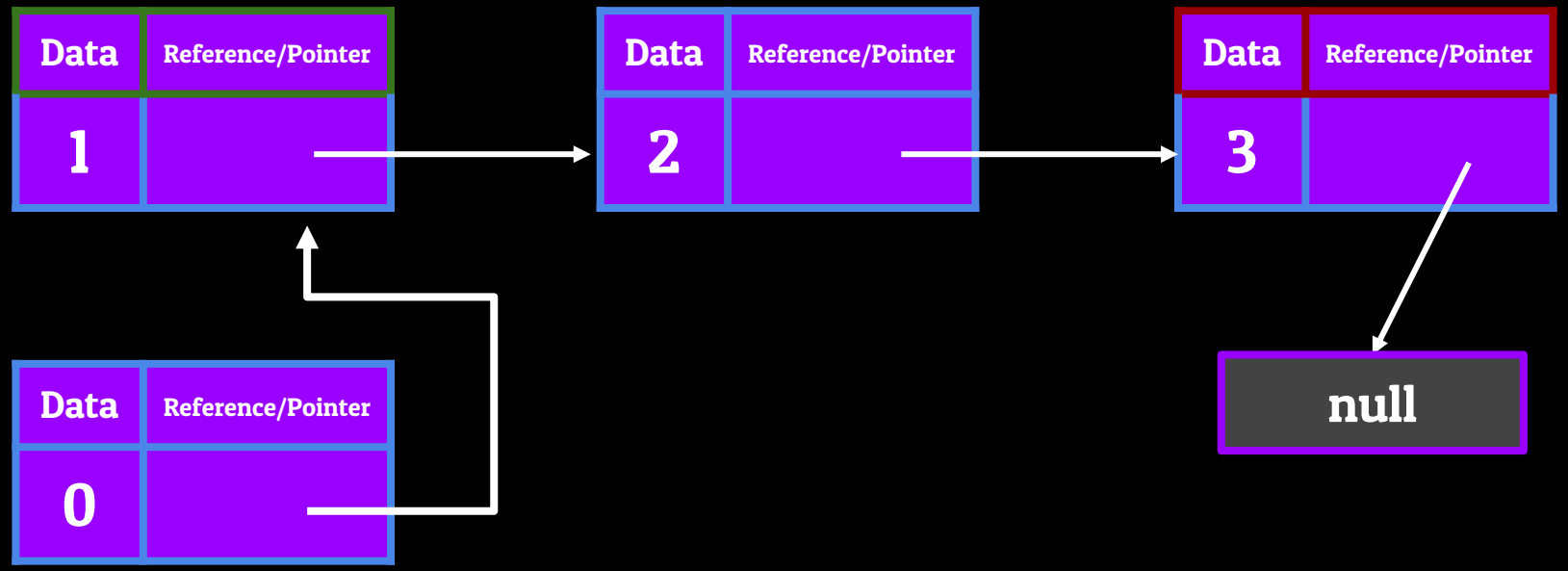
Make that new Node's pointer point to the current Head of the LinkedList



The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

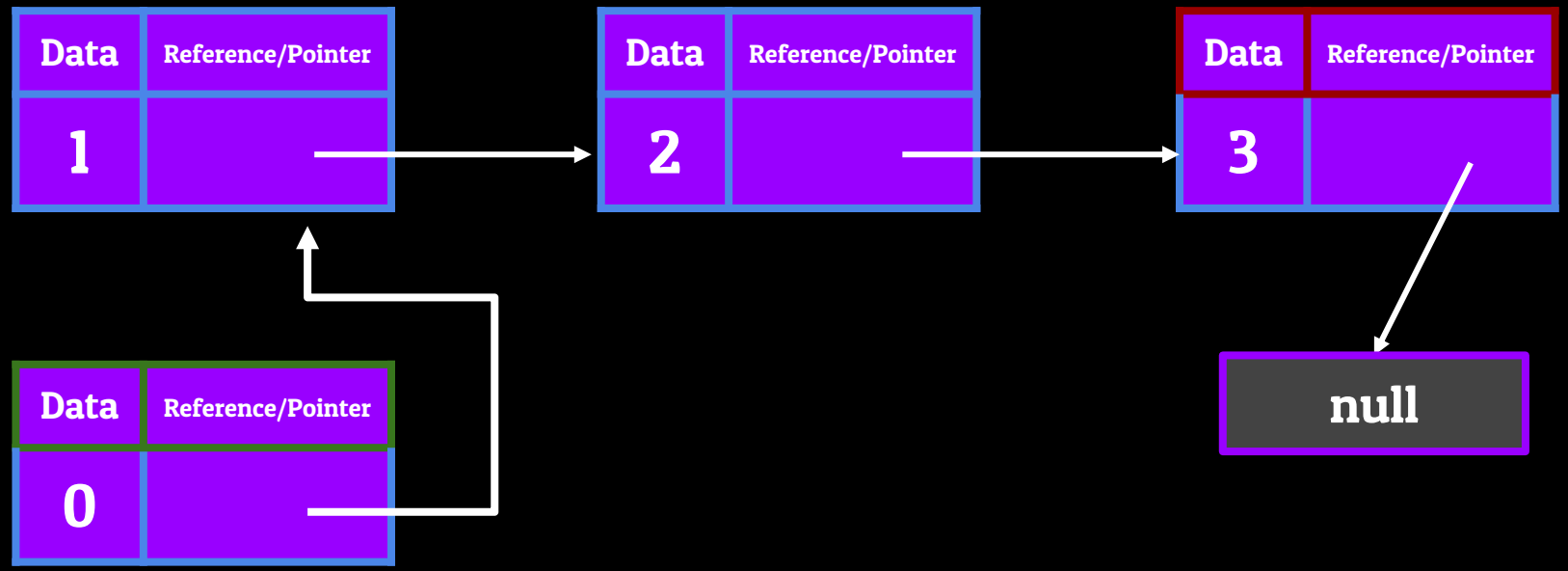
Make that new Node's pointer point to the current Head of the LinkedList



The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

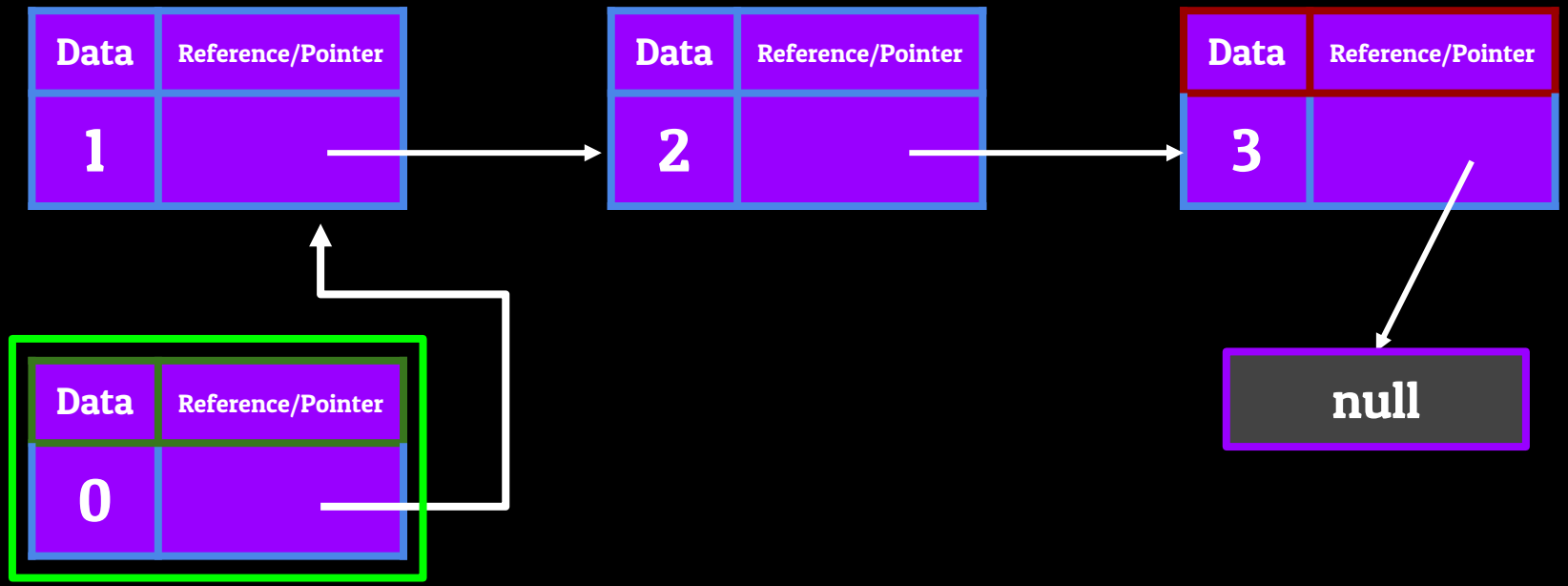
Make that new Node's pointer point to the current Head of the LinkedList



The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

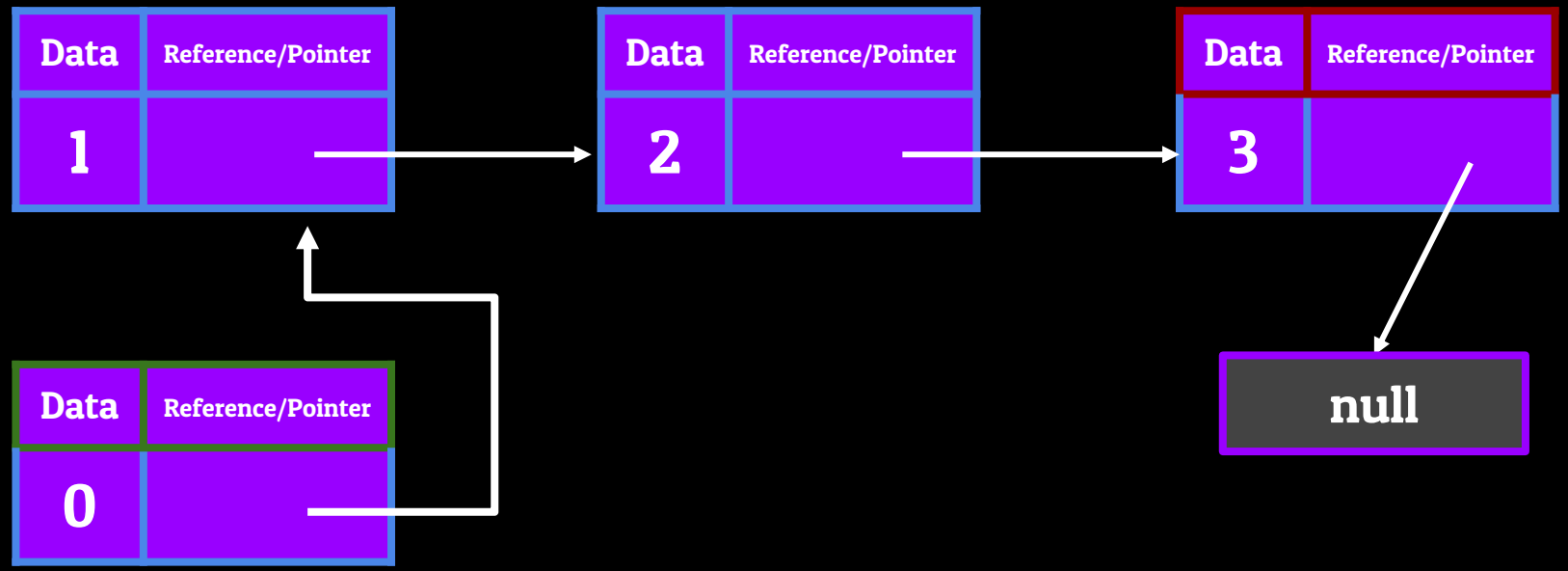
Make that new Node's pointer point to the current Head of the LinkedList



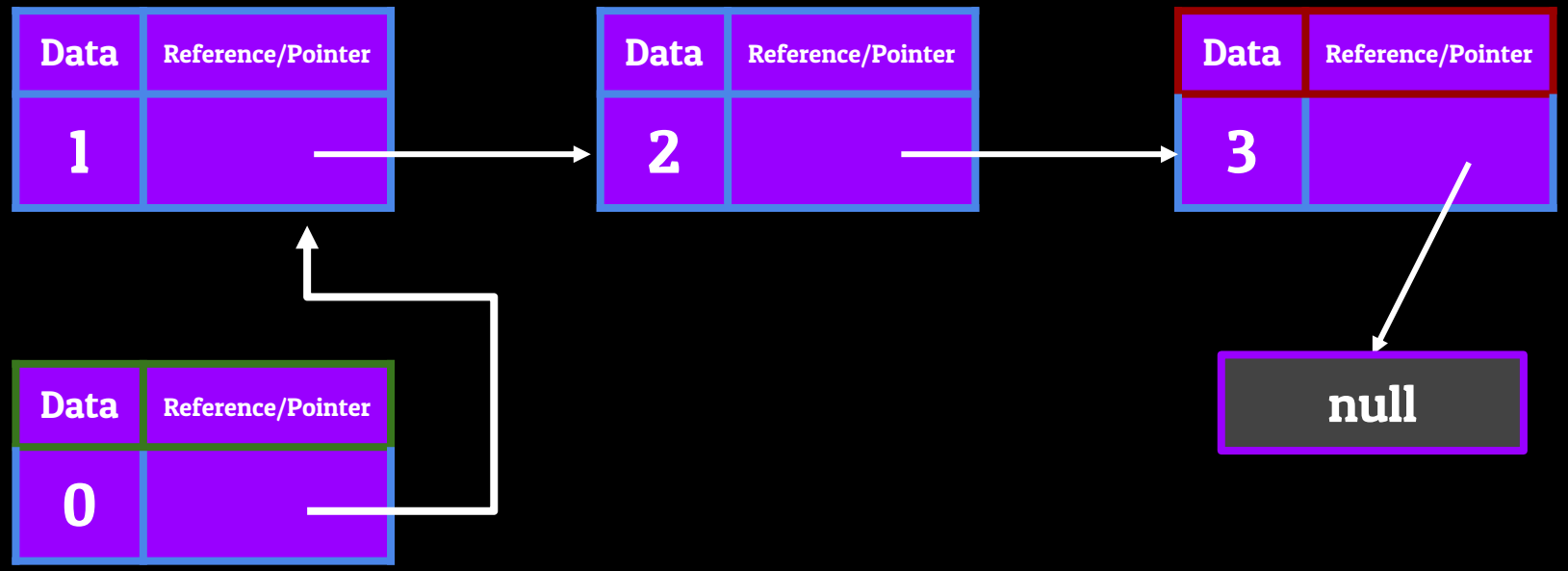
The Linked List - Adding and Removing Information

Adding to the Head of a LinkedList

Make that new Node's pointer point to the current Head of the LinkedList

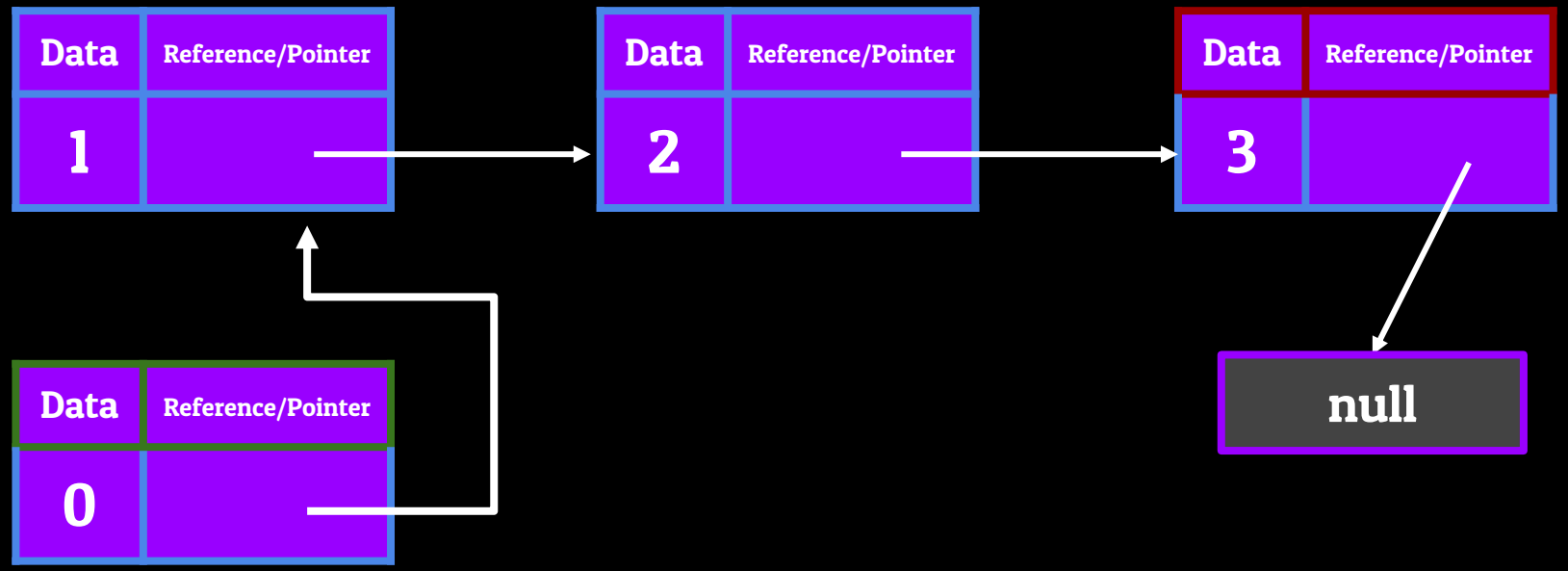


The Linked List - Adding and Removing Information



The Linked List - Adding and Removing Information

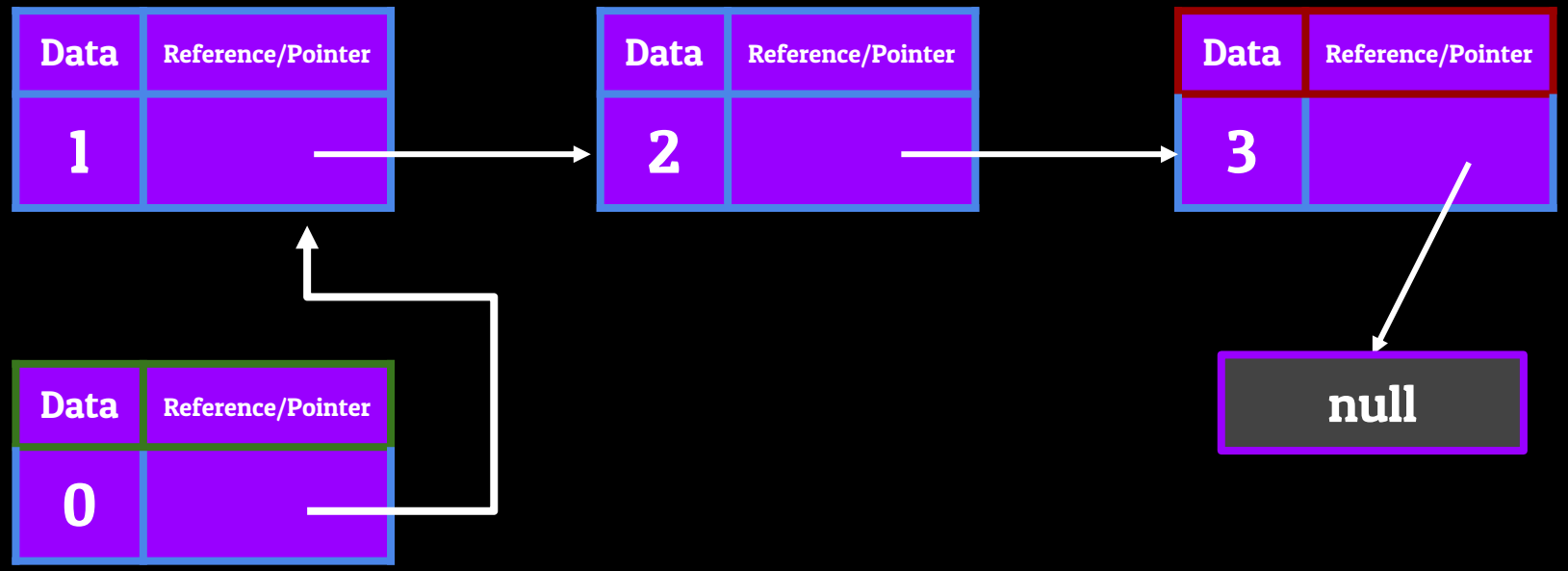
Removing from the Head of a LinkedList



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

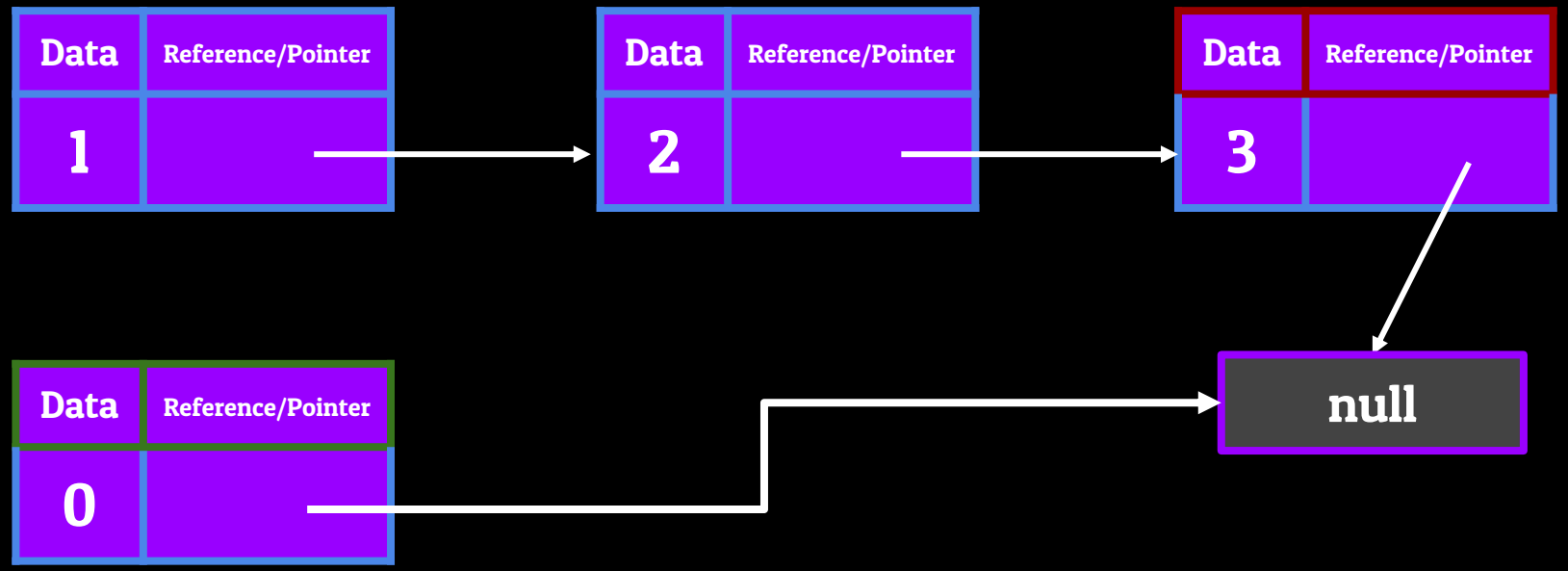
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

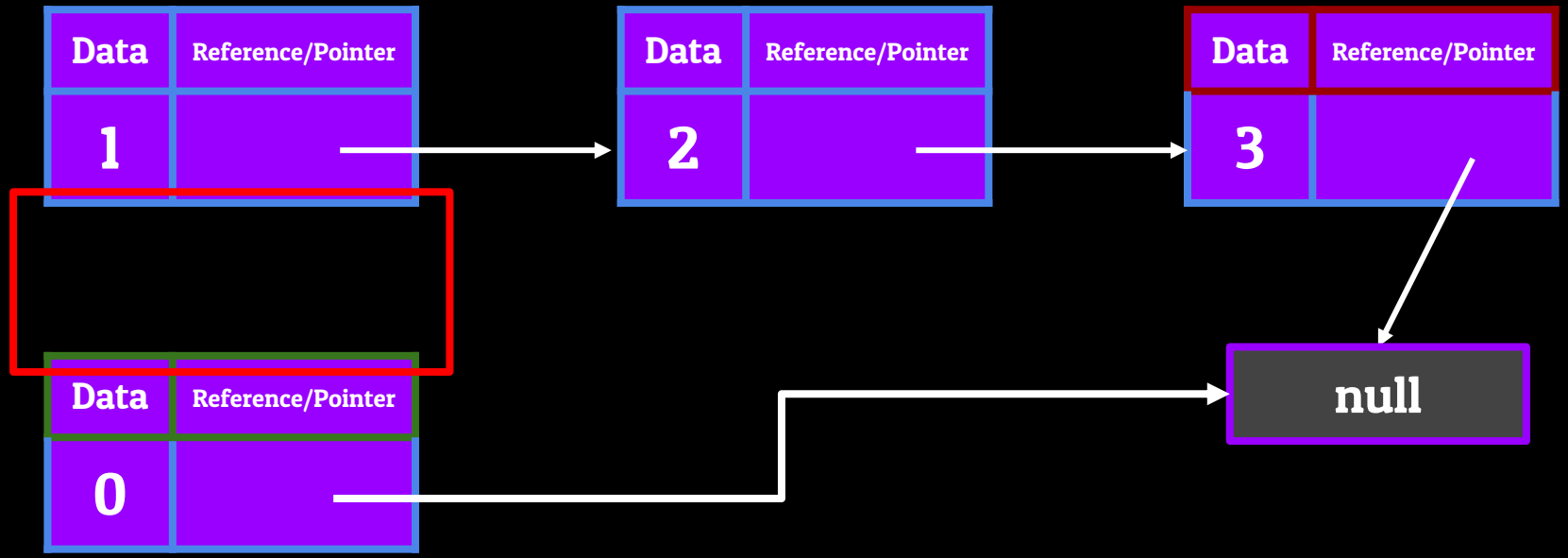
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

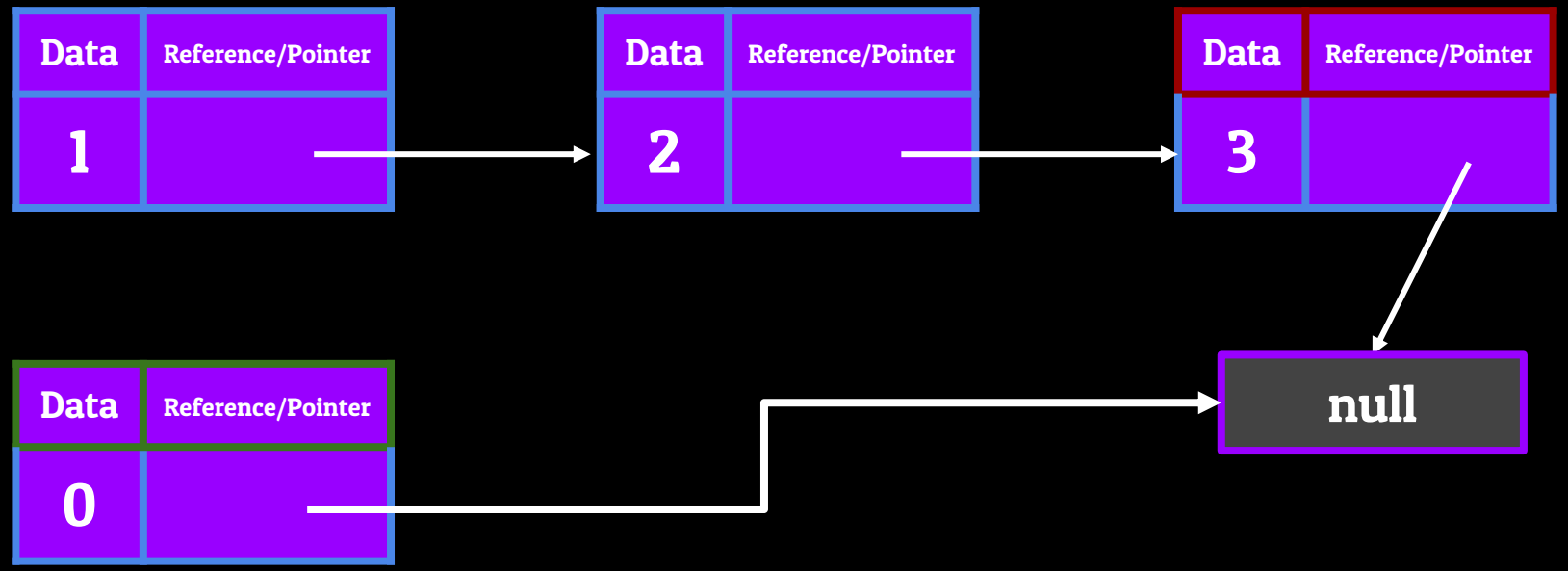
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

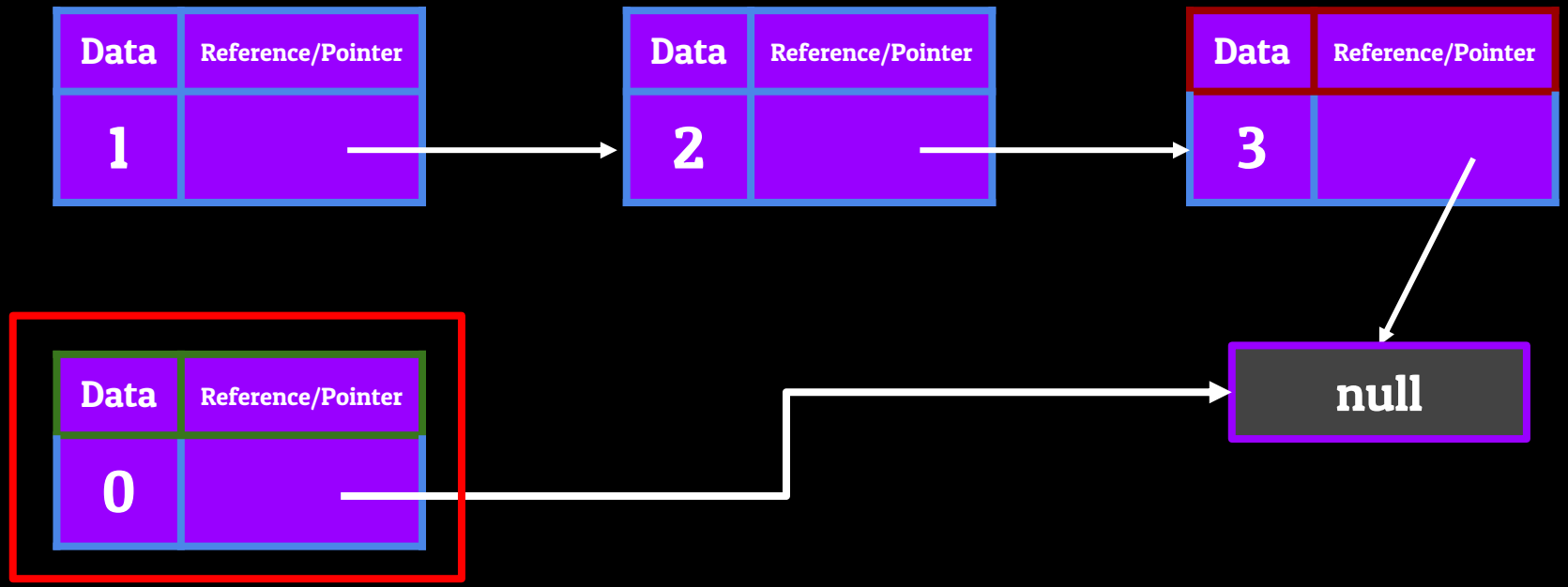
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

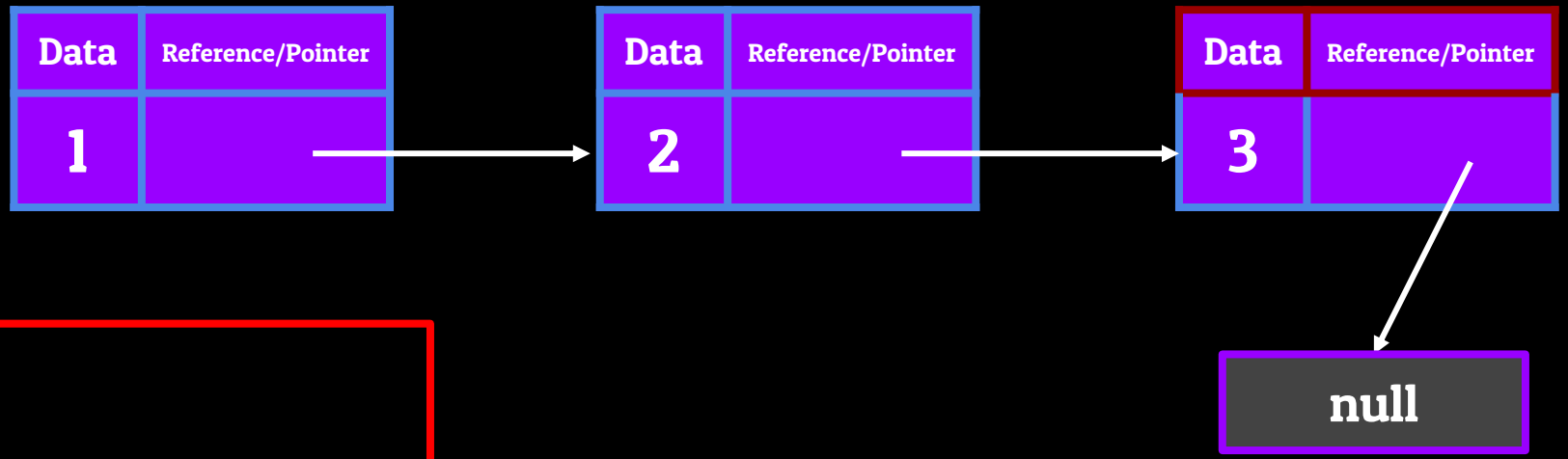
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

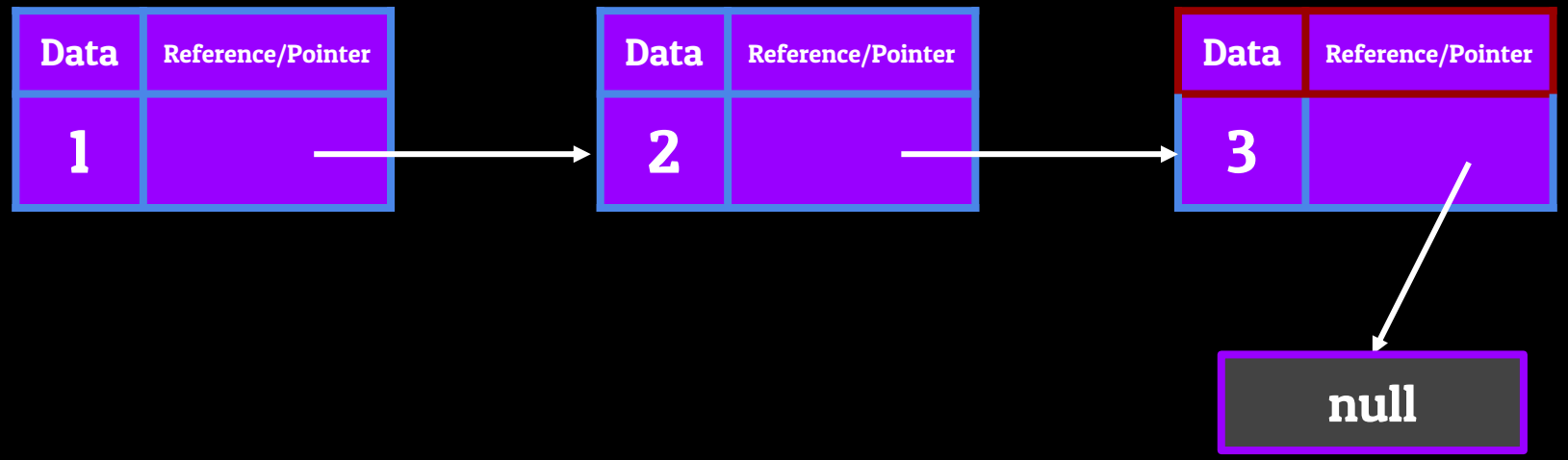
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

Removing from the Head of a LinkedList

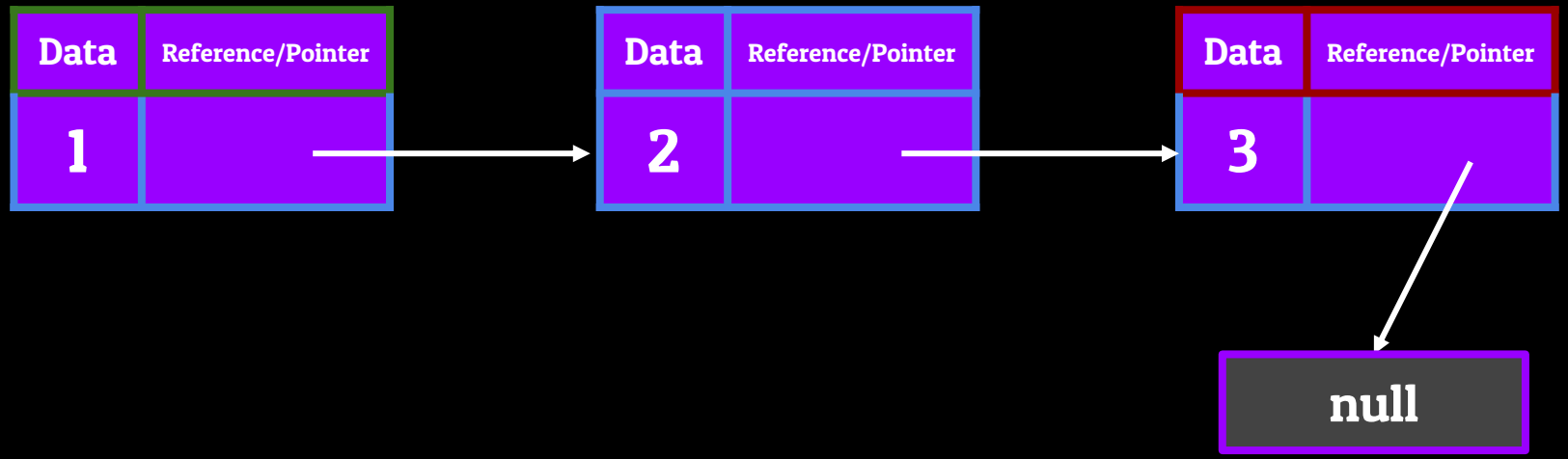
Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

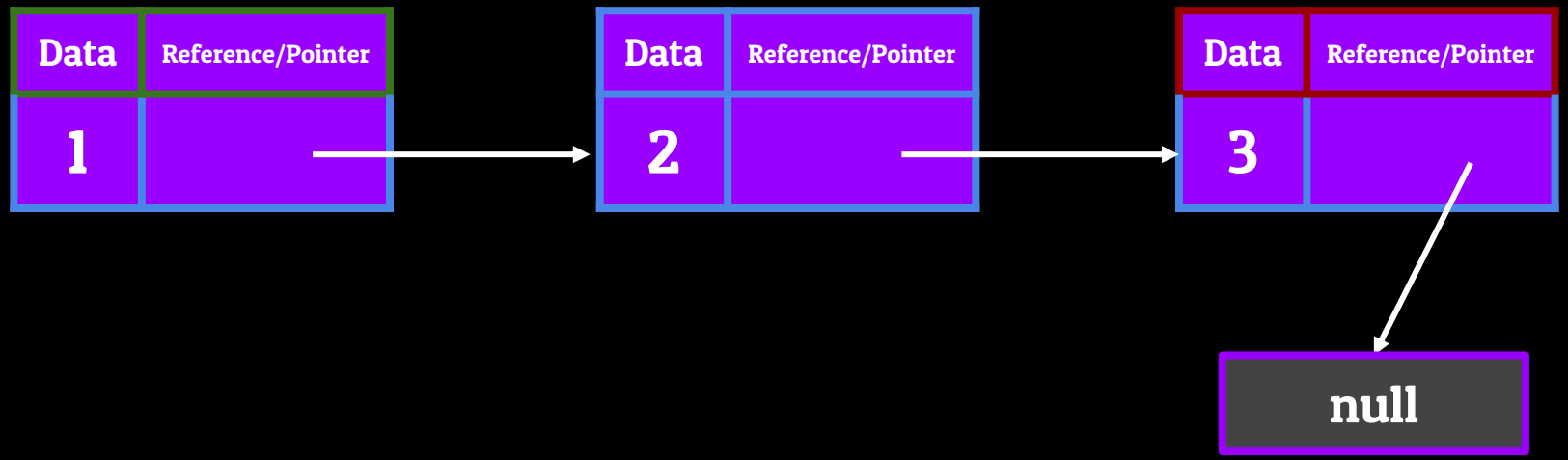
Removing from the Head of a LinkedList

Set the Head Node's pointer to a null value



The Linked List - Adding and Removing Information

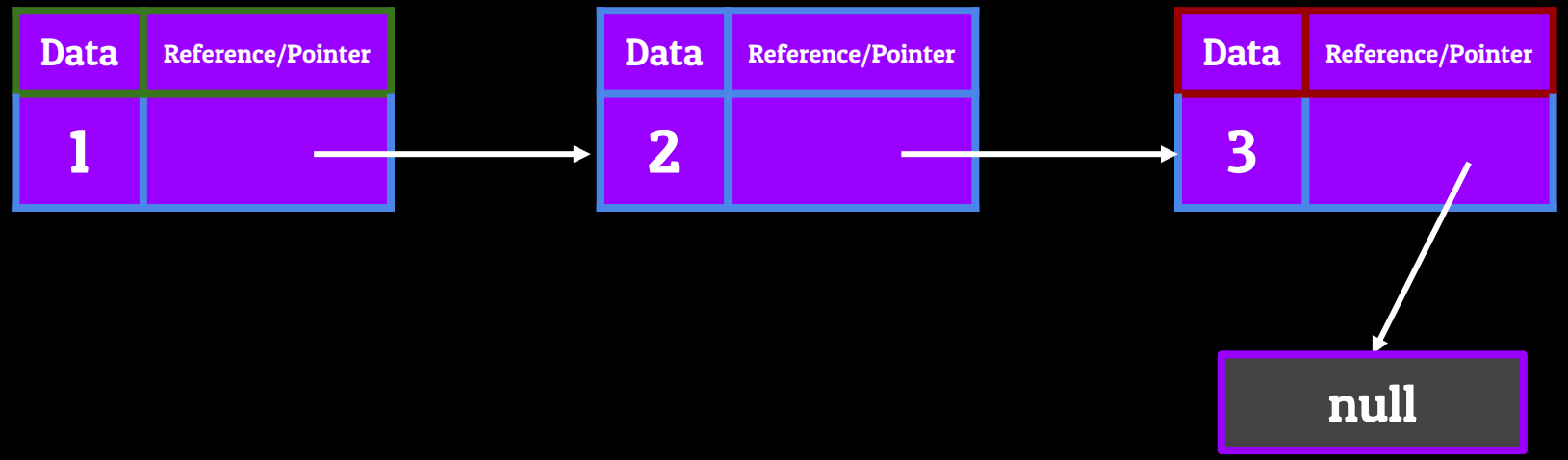
Adding a Node to the Middle of a LinkedList



The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

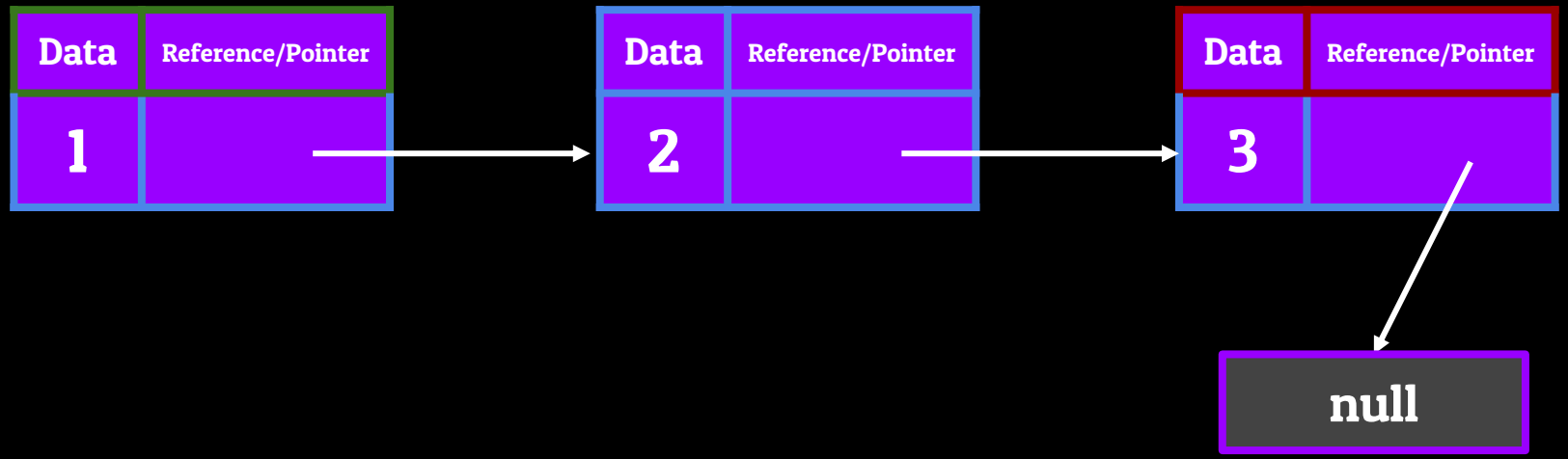


The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node



The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

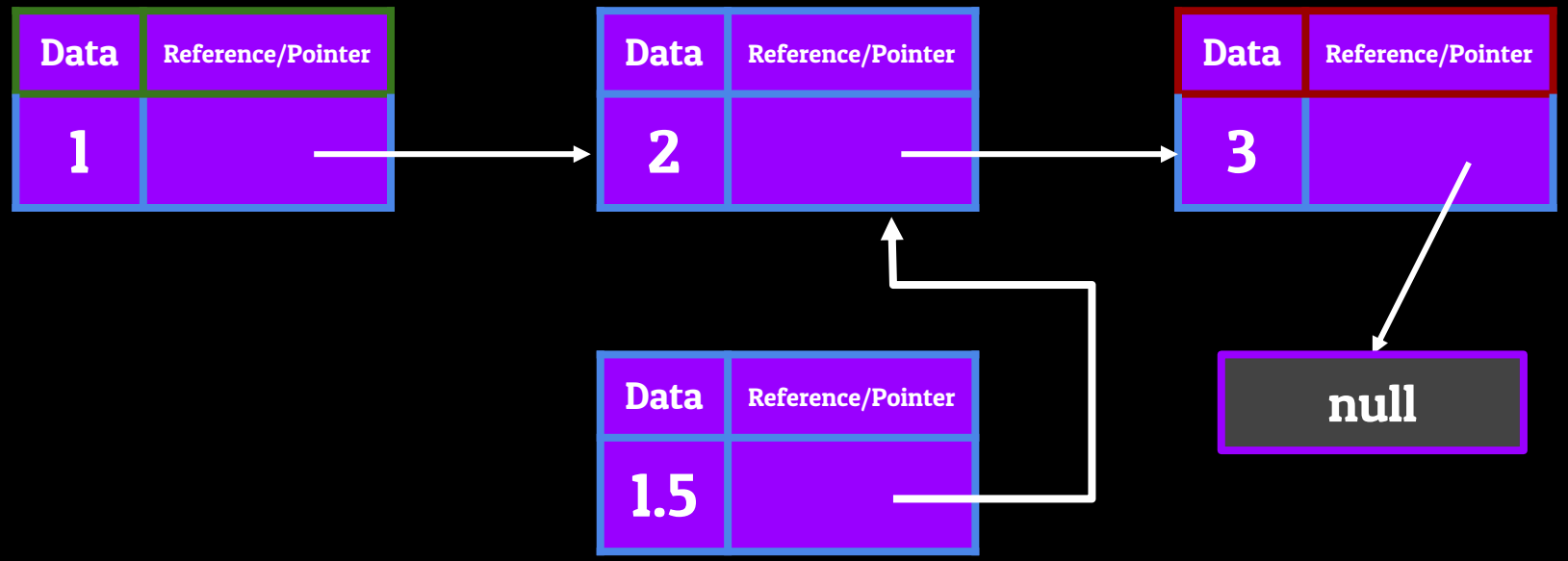
Set the Node before the location we want to insert at to point towards the new Node

The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

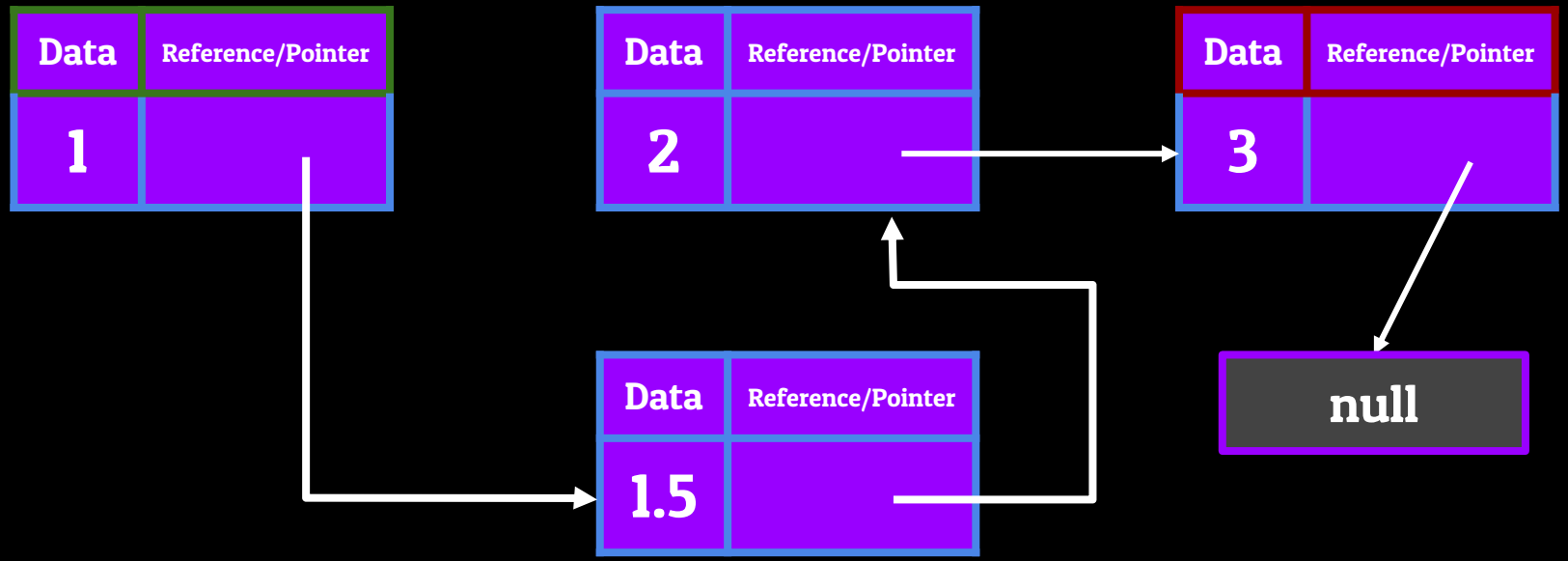


The Linked List - Adding and Removing Information

Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

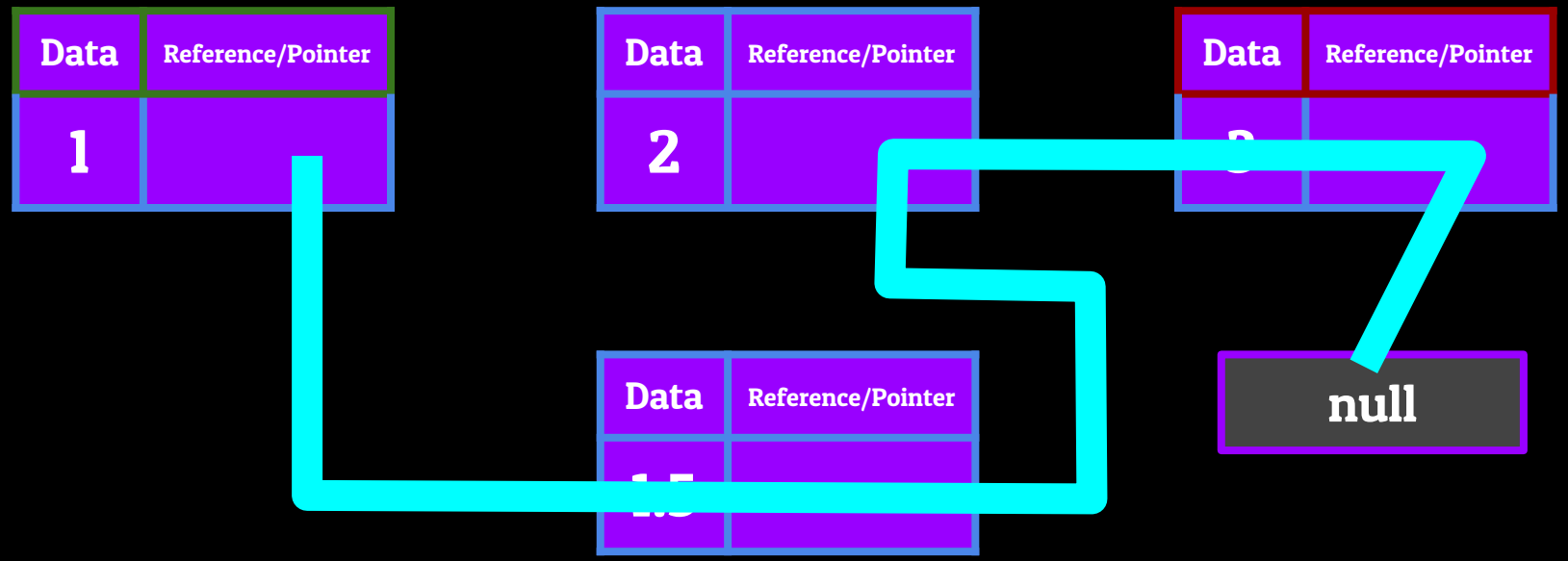


The Linked List - Adding and Removing Information

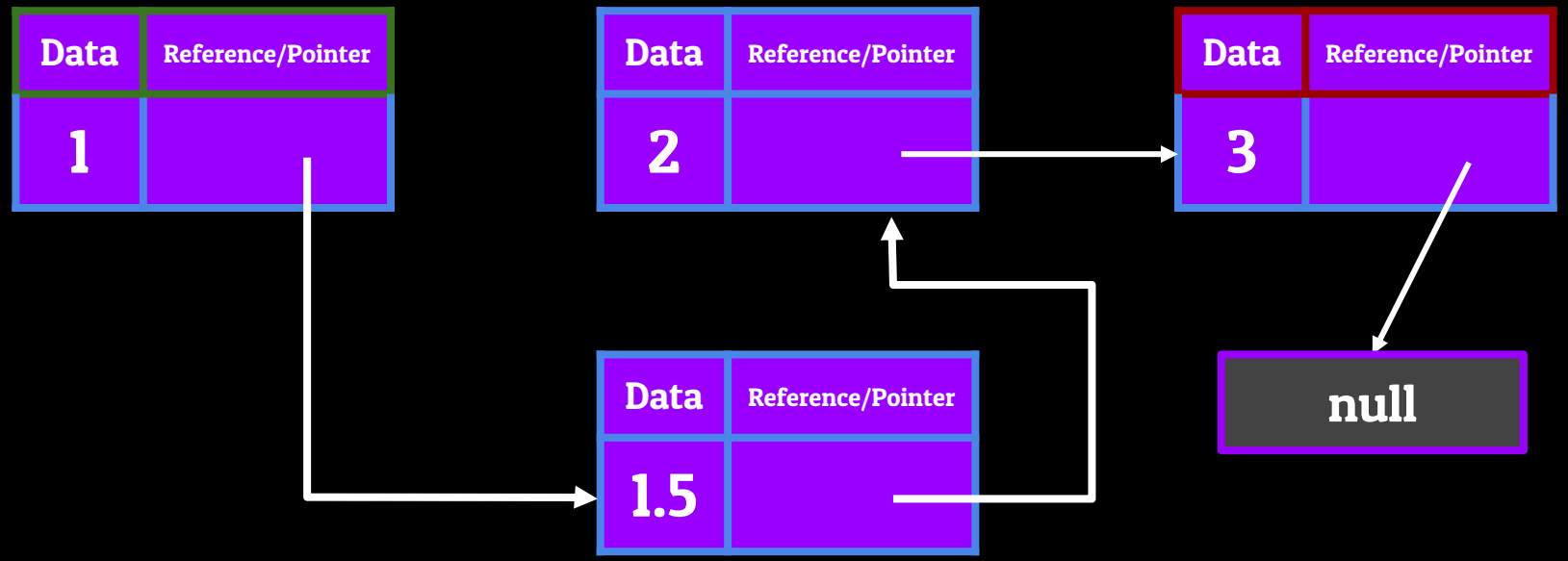
Adding a Node to the Middle of a LinkedList

Make the pointer of the new Node point to the Node after the location we want to insert at

Set the Node before the location we want to insert at to point towards the new Node

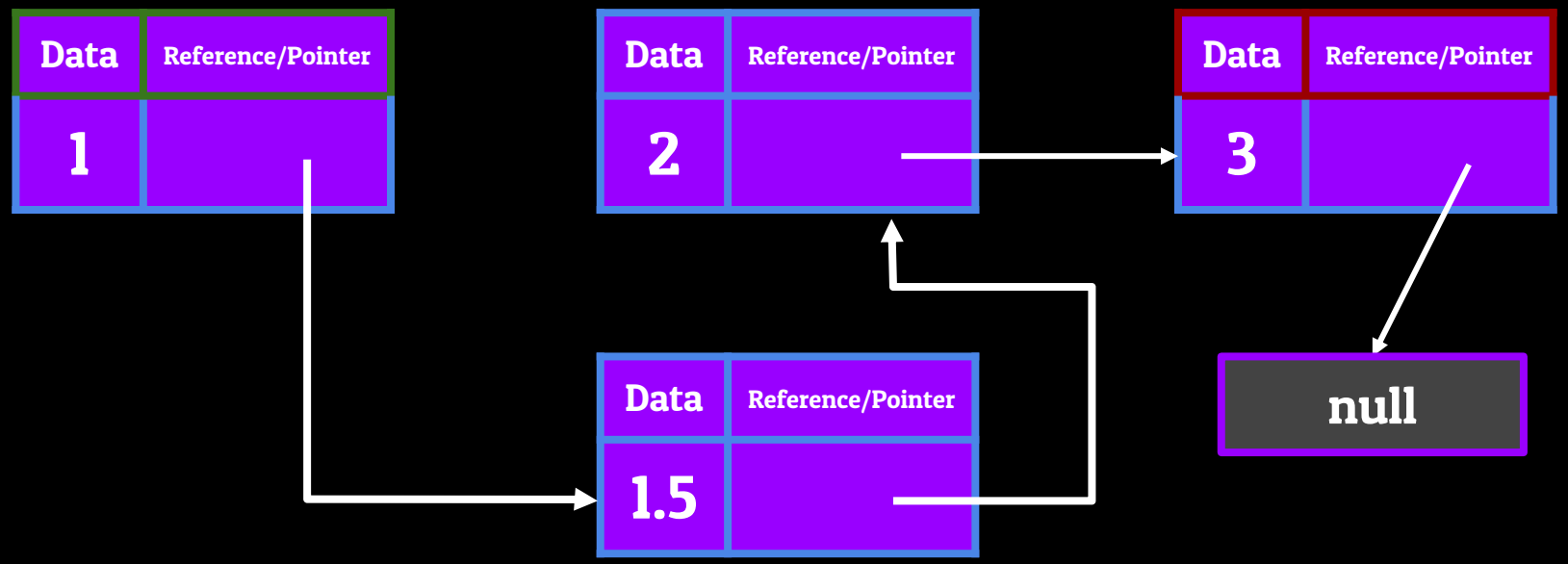


The Linked List - Adding and Removing Information



The Linked List - Adding and Removing Information

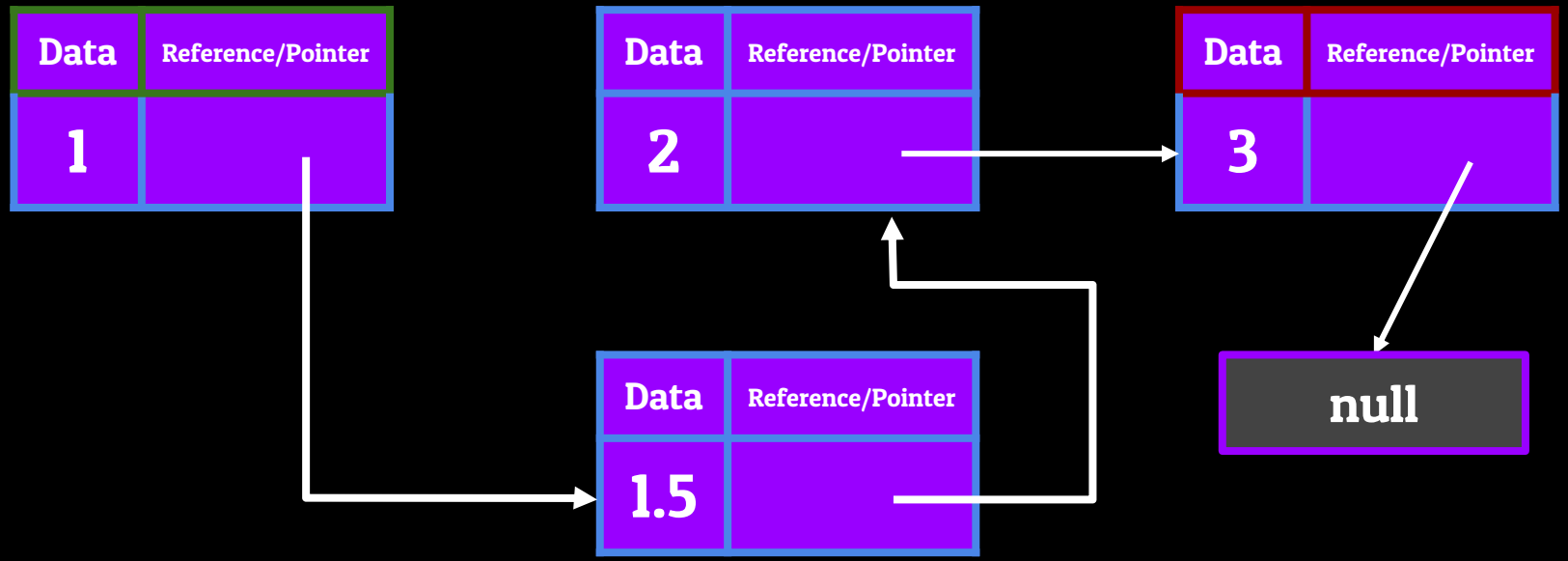
Removing a Node from the middle of a LinkedList



The Linked List - Adding and Removing Information

Removing a Node from the middle of a LinkedList

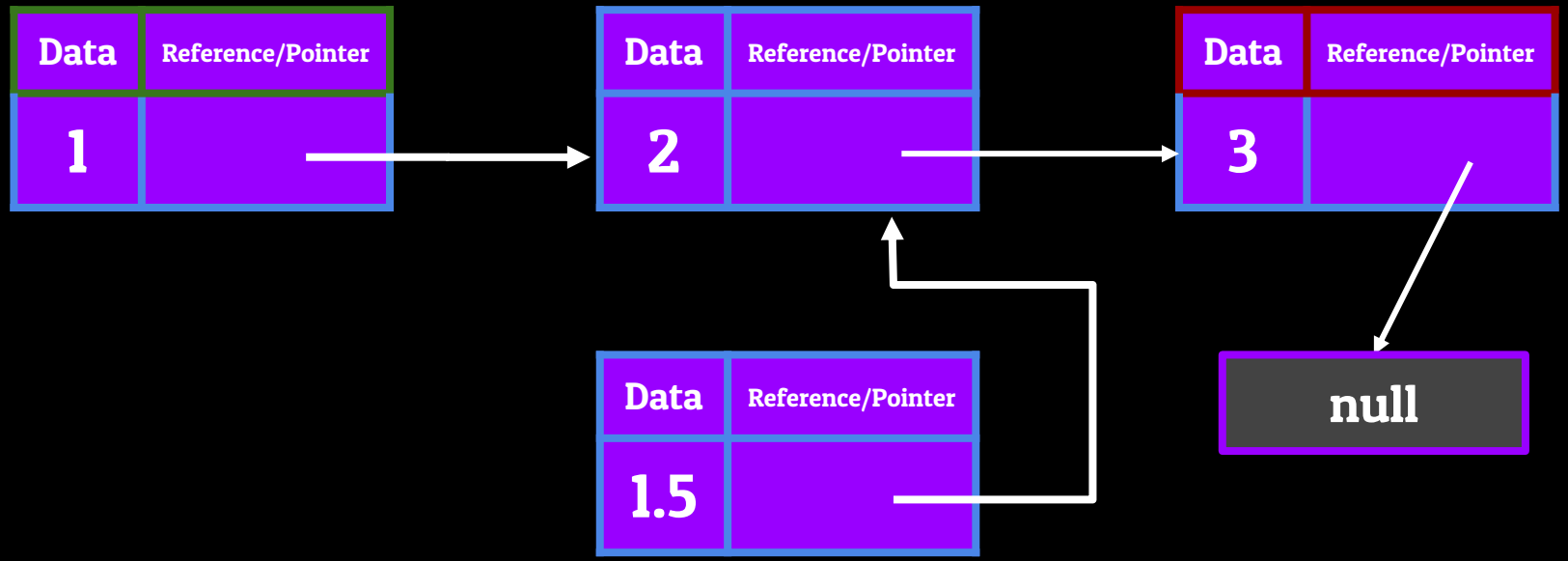
Make the pointer of the Node previous to the one we're removing, to now point to the Node after the one we're removing



The Linked List - Adding and Removing Information

Removing a Node from the middle of a LinkedList

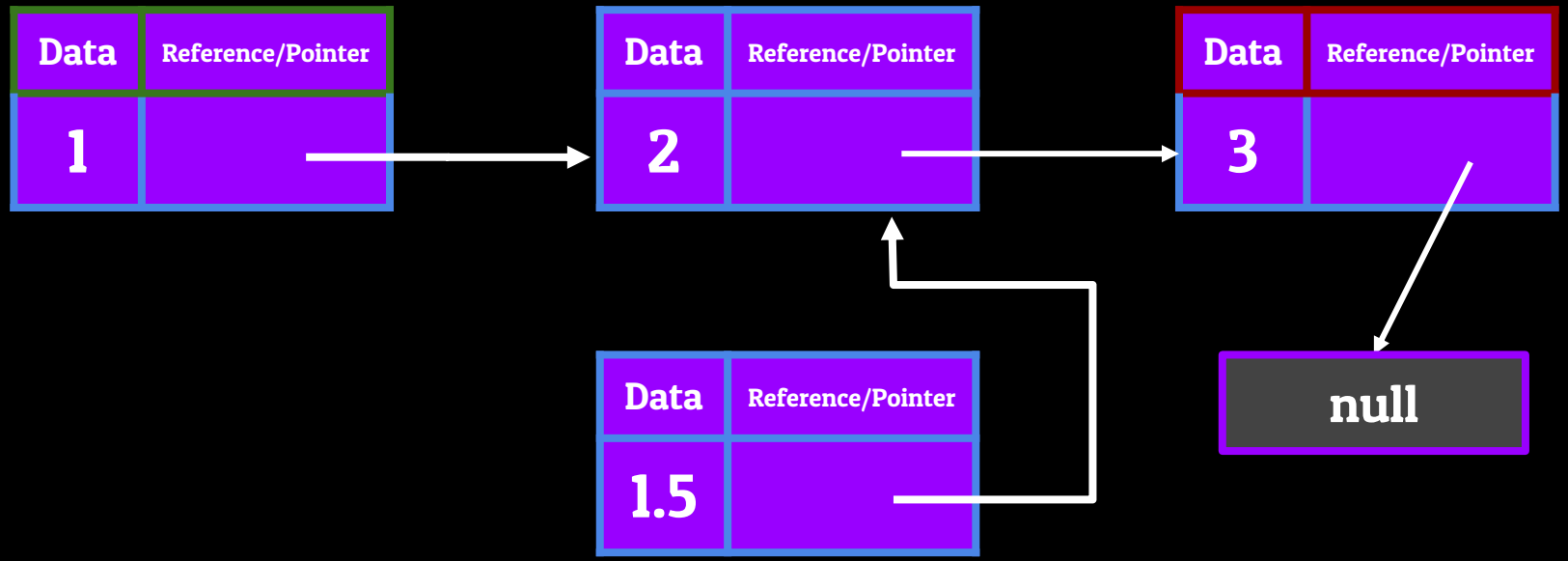
Make the pointer of the Node previous to the one we're removing, to now point to the Node after the one we're removing



The Linked List - Adding and Removing Information

Removing a Node from the middle of a LinkedList

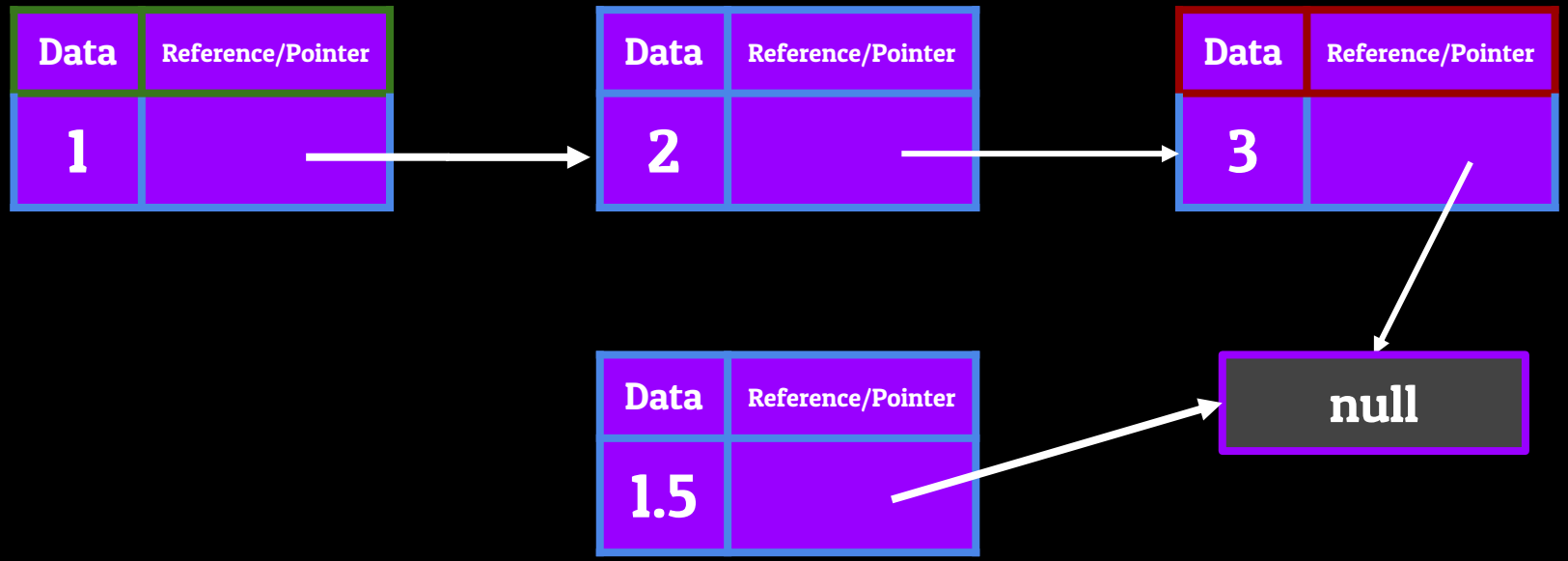
Make the pointer of the Node previous to the one we're removing, to now point to the Node after the one we're removing



The Linked List - Adding and Removing Information

Removing a Node from the middle of a LinkedList

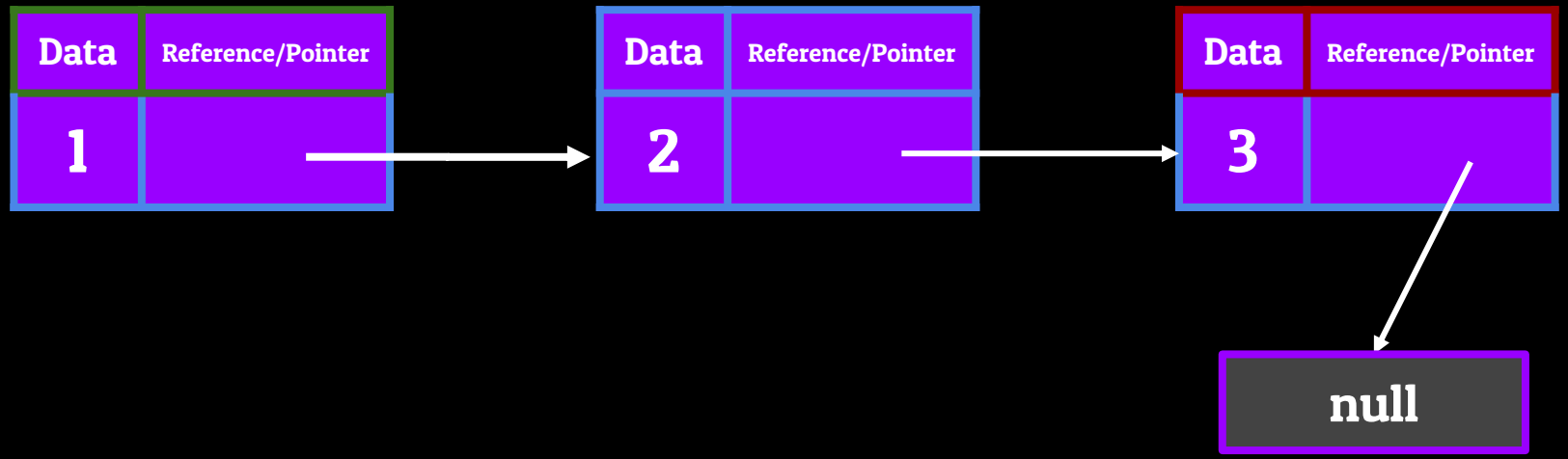
Make the pointer of the Node previous to the one we're removing, to now point to the Node after the one we're removing



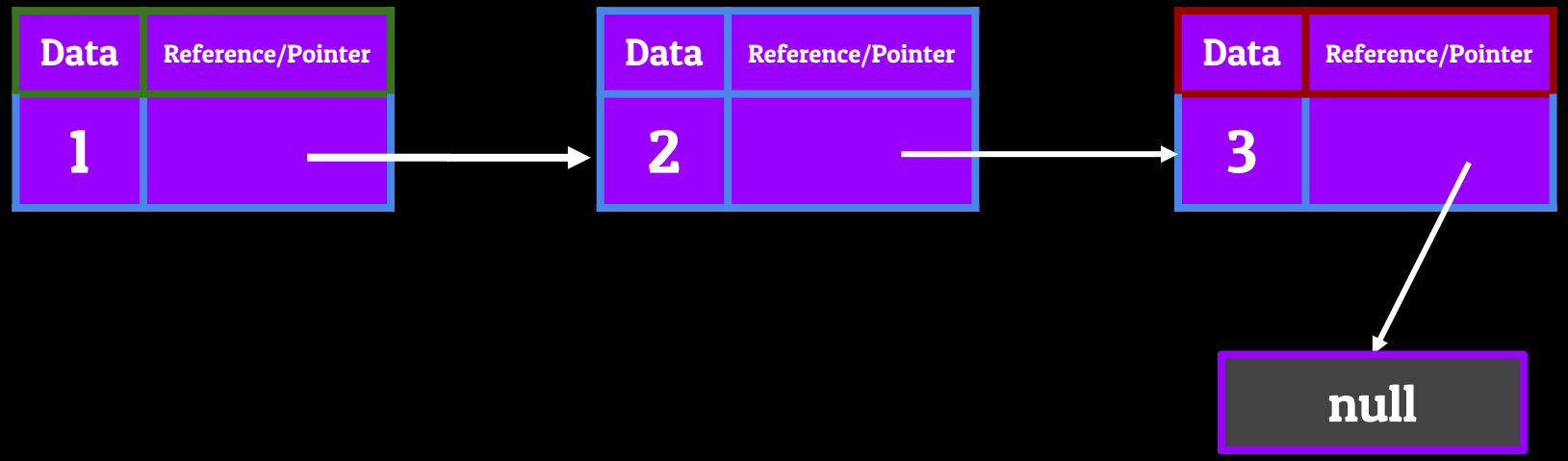
The Linked List - Adding and Removing Information

Removing a Node from the middle of a LinkedList

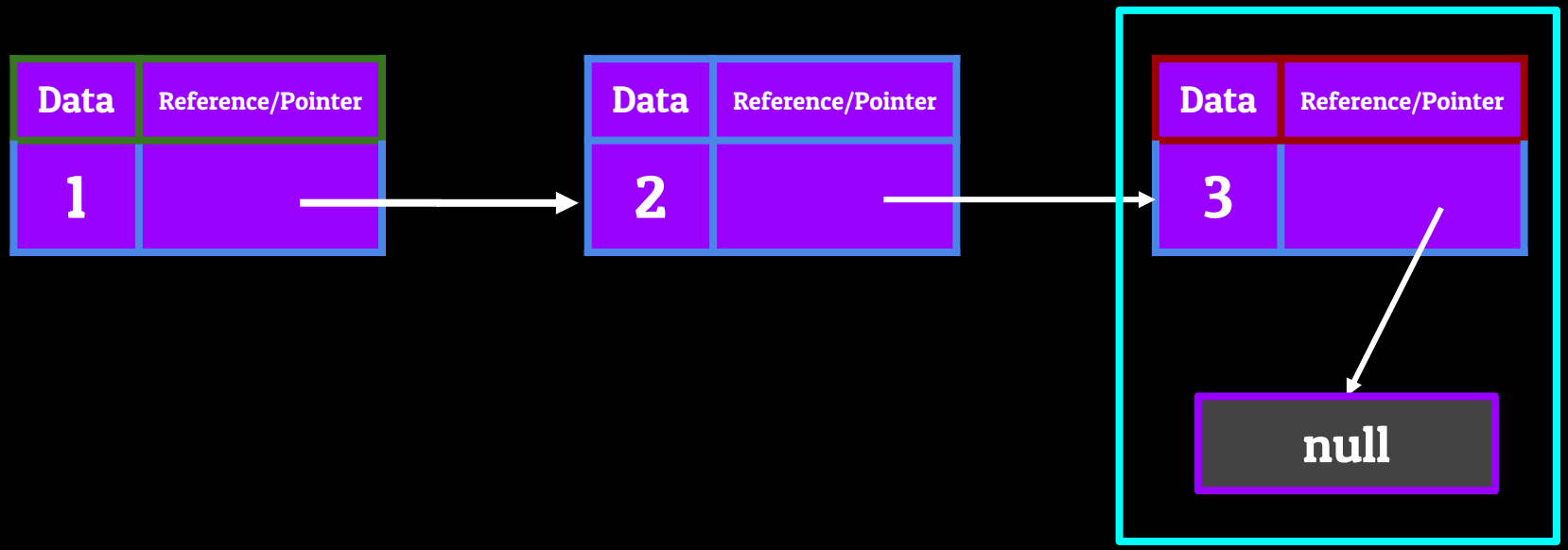
Make the pointer of the Node previous to the one we're removing, to now point to the Node after the one we're removing



The Linked List - Adding and Removing Information

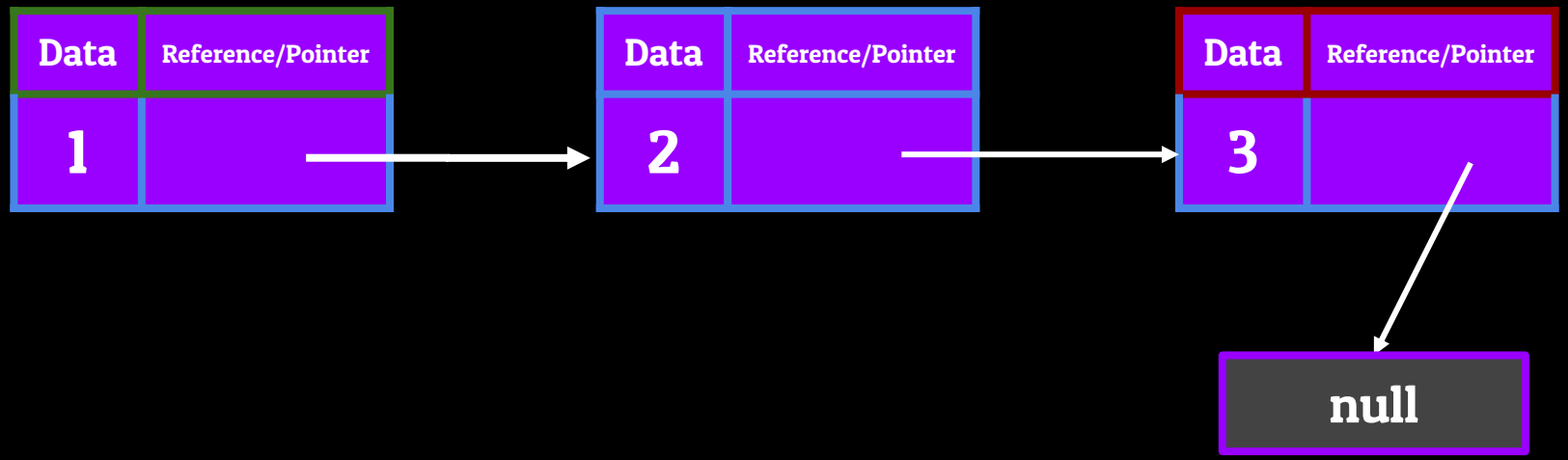


The Linked List - Adding and Removing Information



The Linked List - Adding and Removing Information

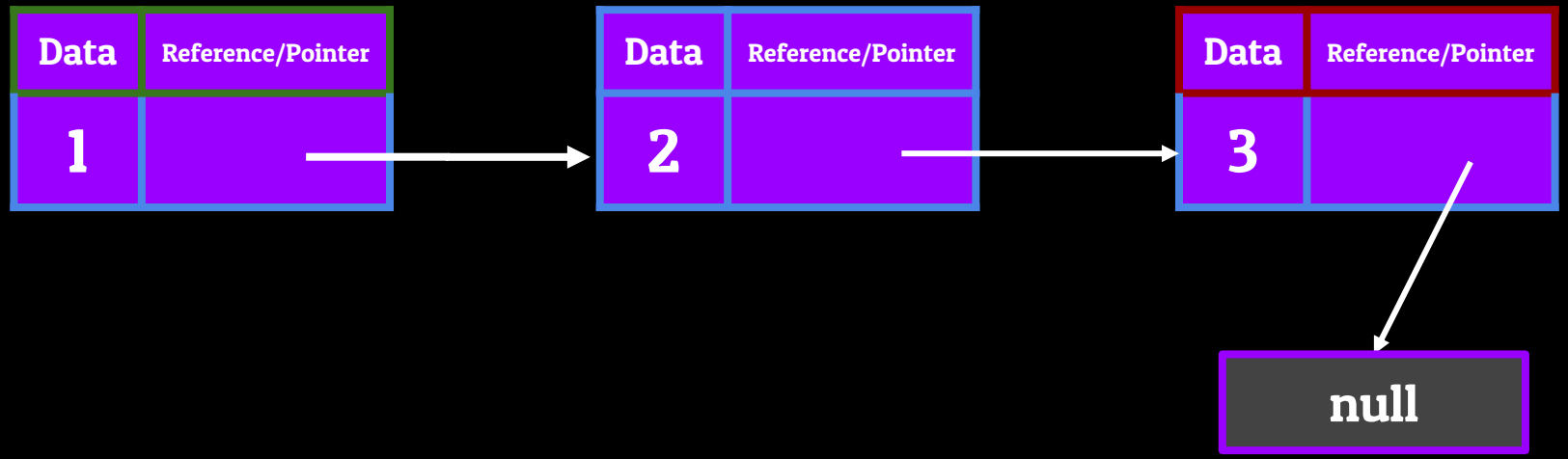
Adding to the Tail of a LinkedList



The Linked List - Adding and Removing Information

Adding to the Tail of a LinkedList

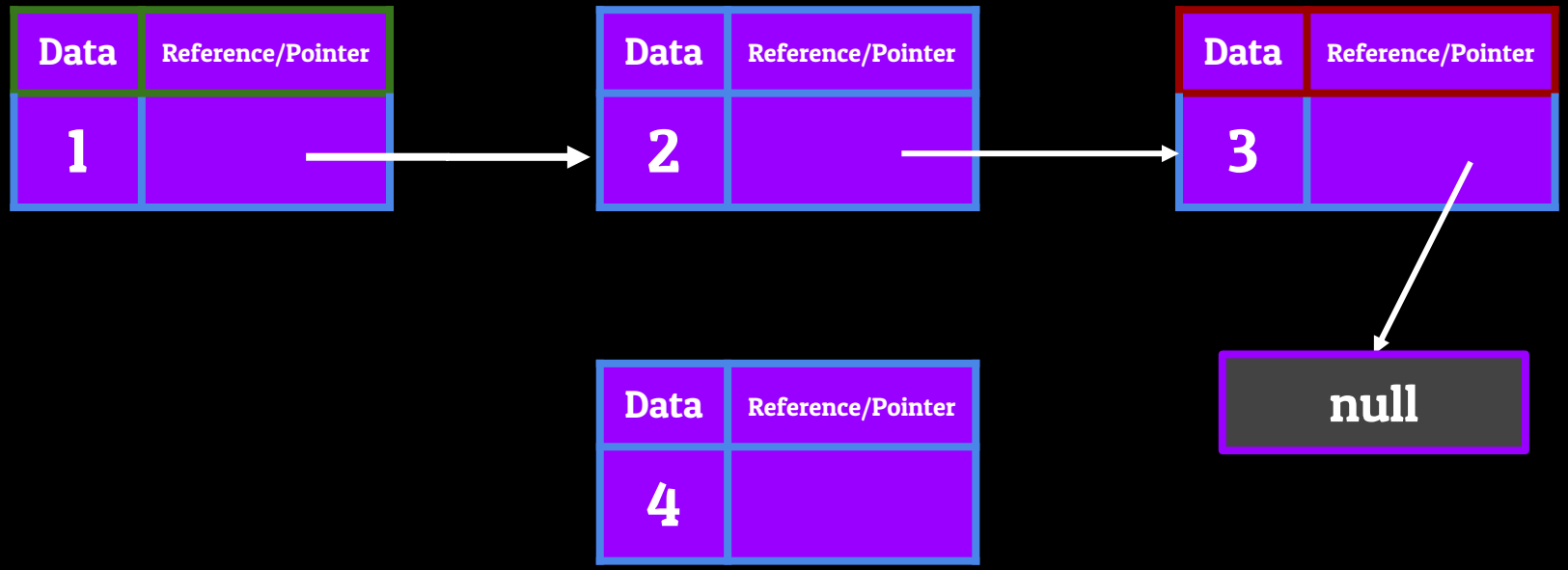
Make the current tail point towards the new Node you want to add



The Linked List - Adding and Removing Information

Adding to the Tail of a LinkedList

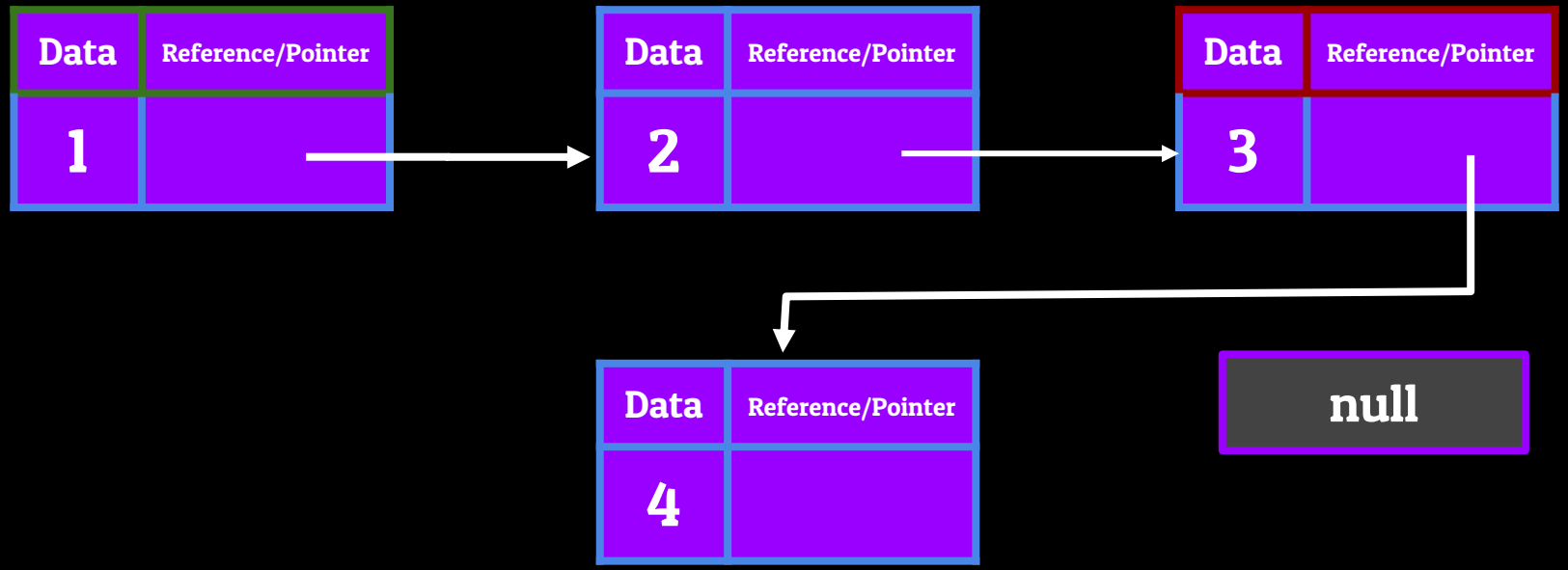
Make the current tail point towards the new Node you want to add



The Linked List - Adding and Removing Information

Adding to the Tail of a LinkedList

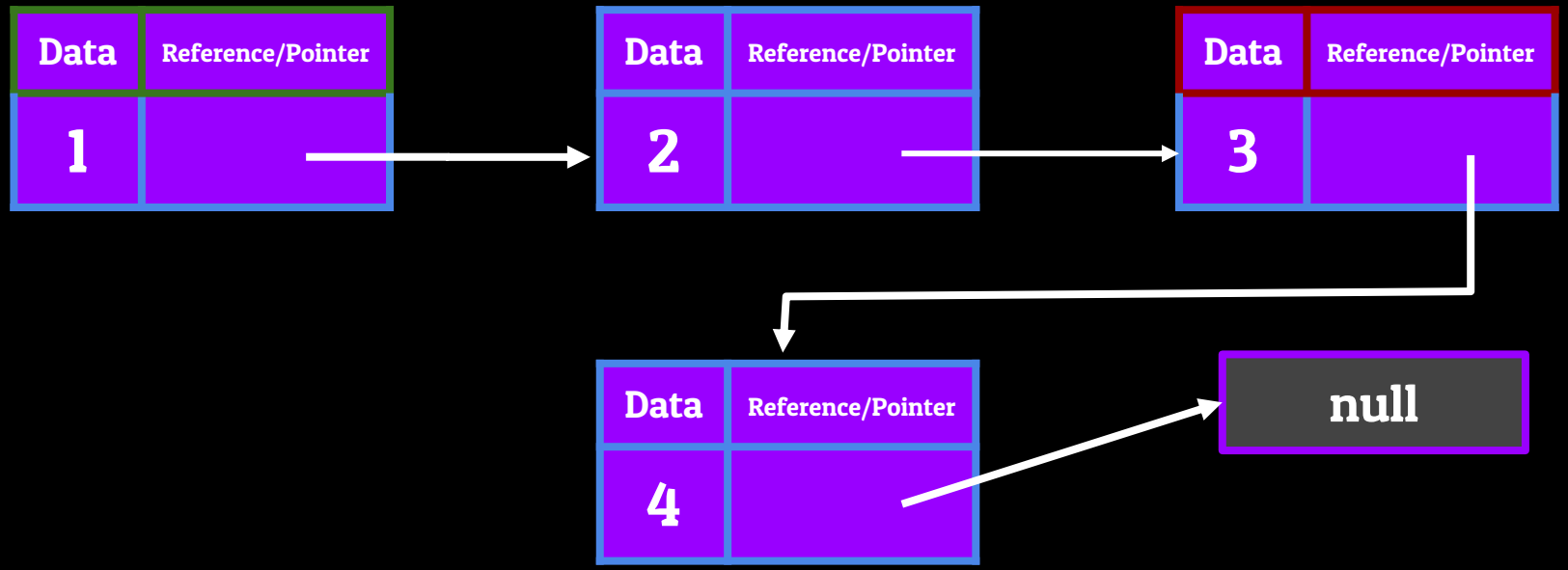
Make the current tail point towards the new Node you want to add



The Linked List - Adding and Removing Information

Adding to the Tail of a LinkedList

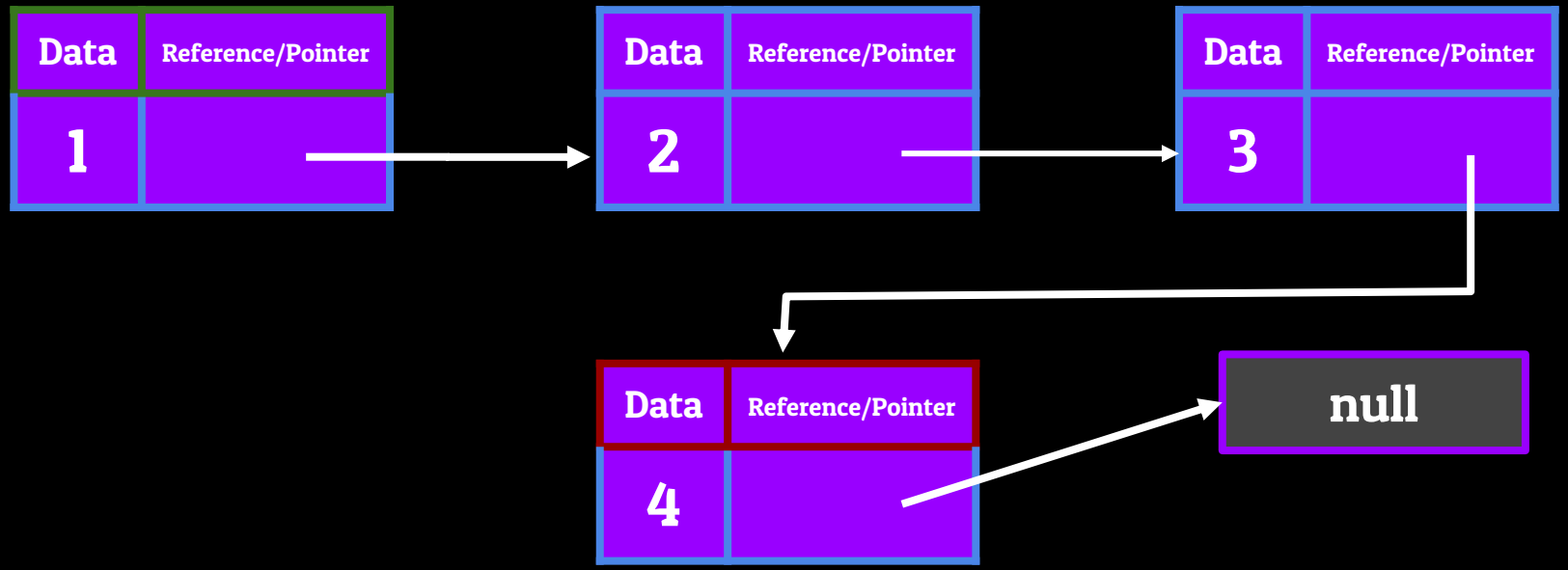
Make the current tail point towards the new Node you want to add



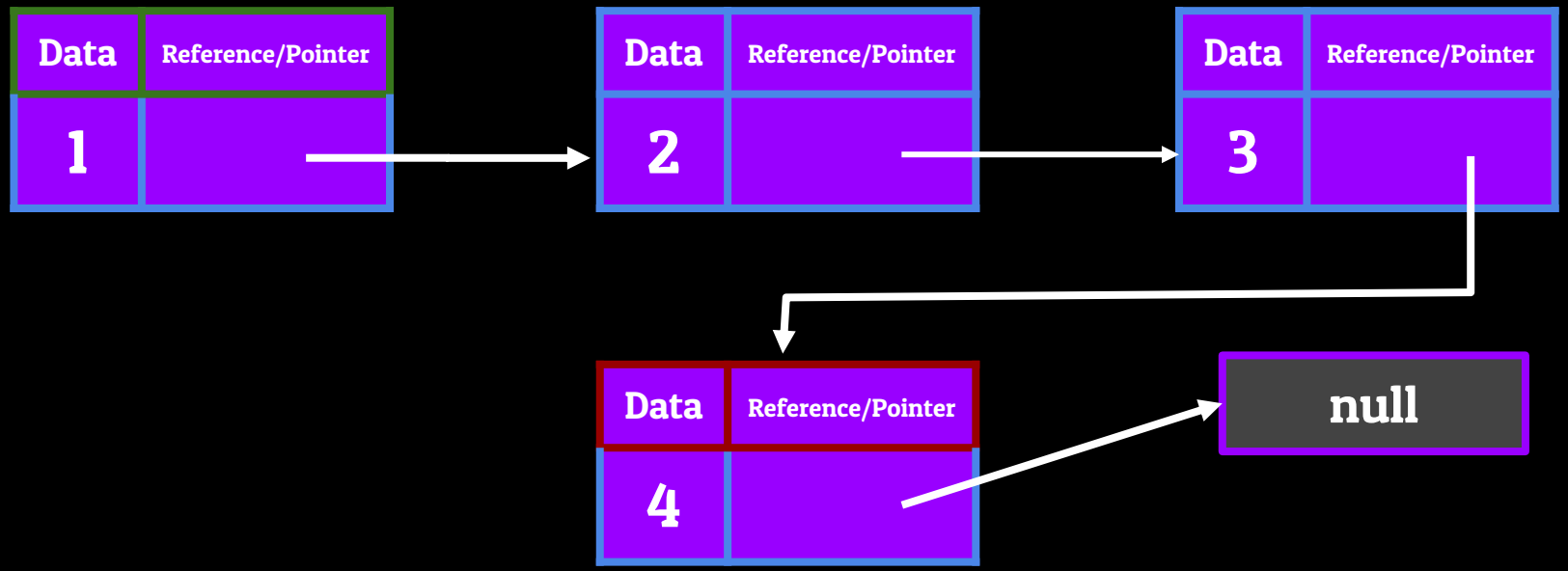
The Linked List - Adding and Removing Information

Adding to the Tail of a LinkedList

Make the current tail point towards the new Node you want to add

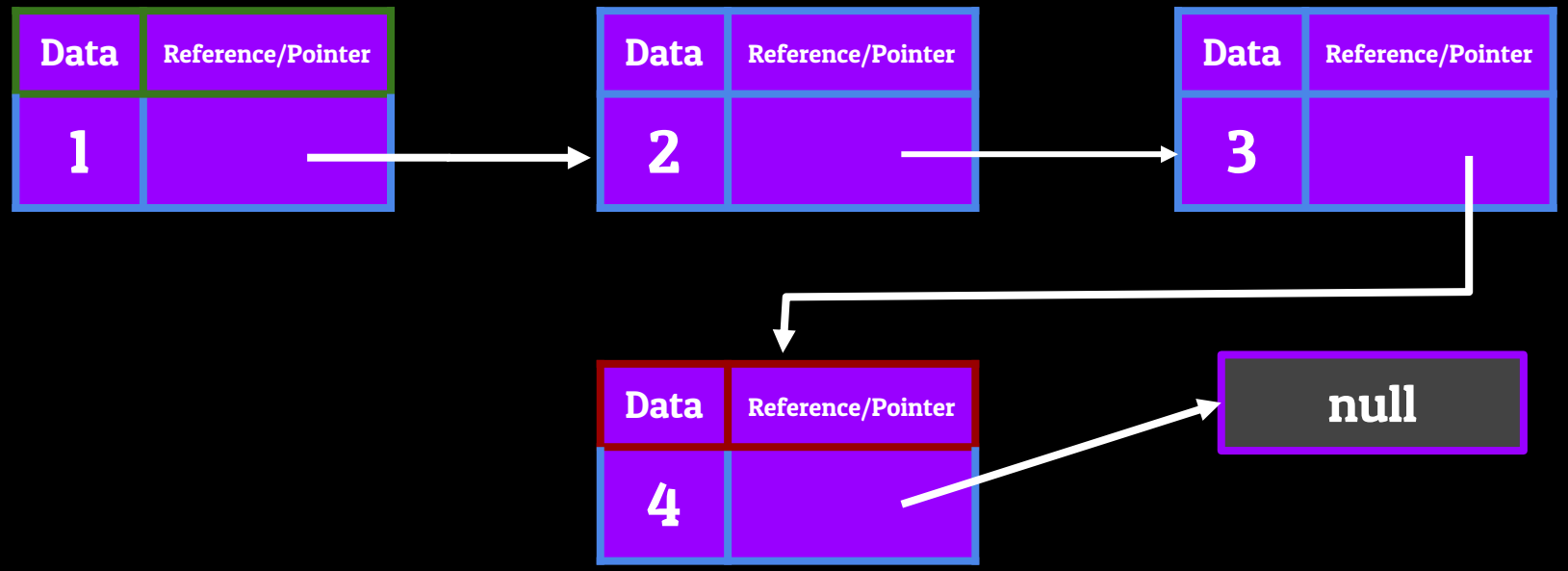


The Linked List - Adding and Removing Information



The Linked List - Adding and Removing Information

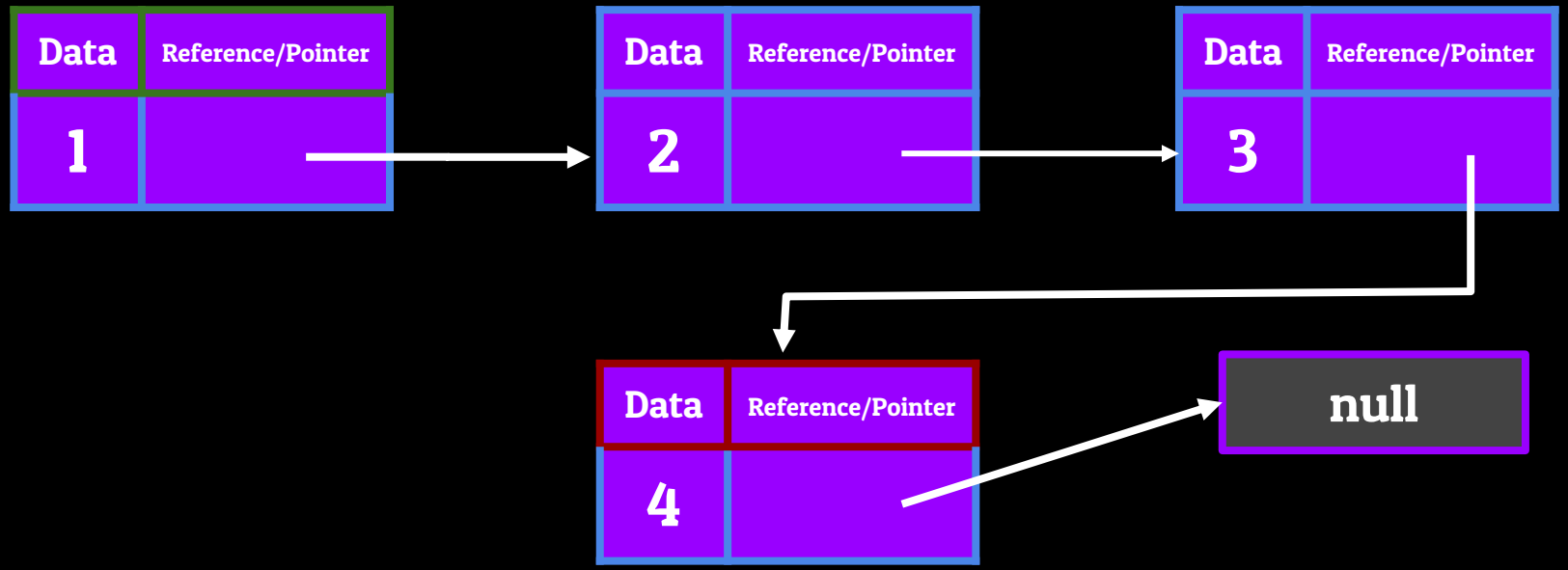
Removing from the Tail of a LinkedList



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

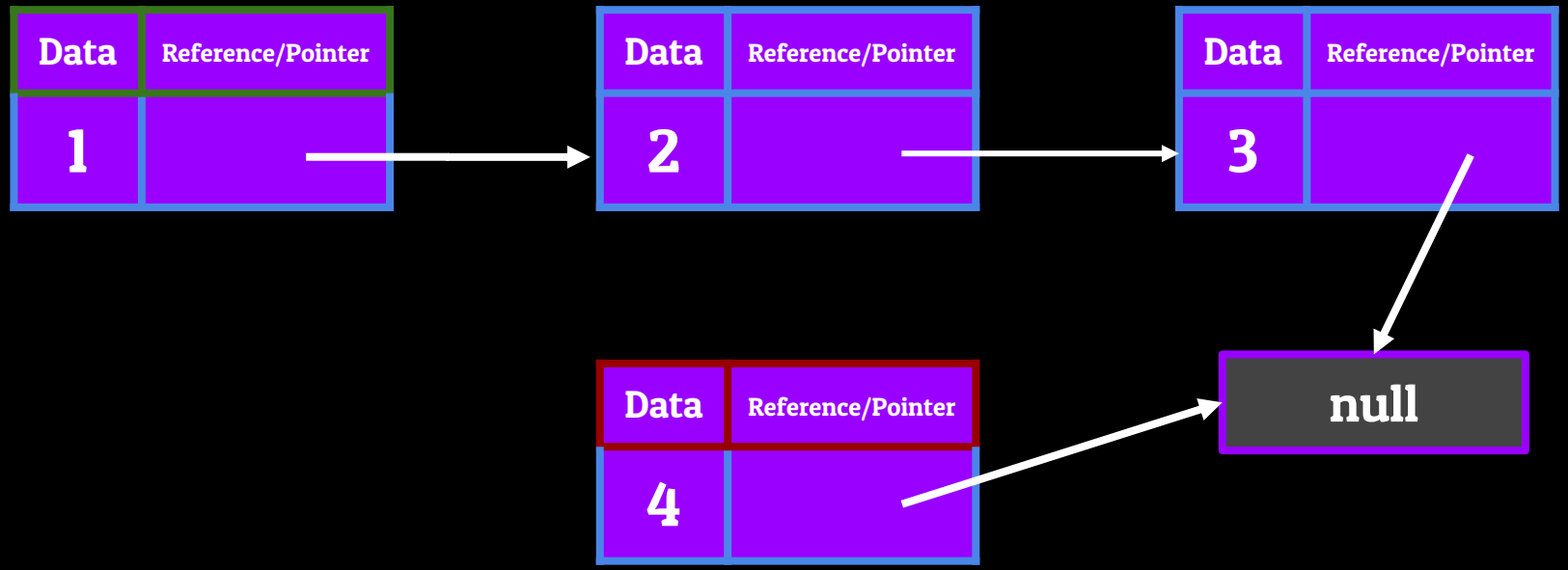
Set the previous tail to point towards a null value instead of the current tail



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

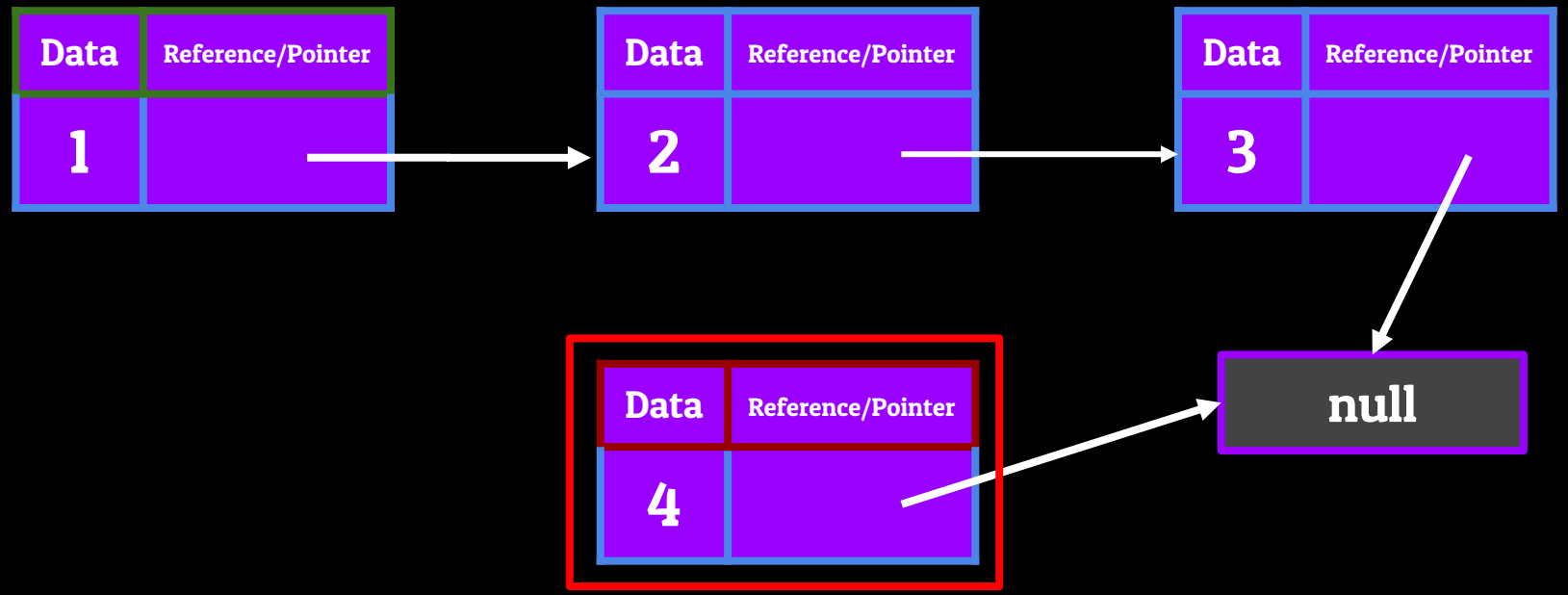
Set the previous tail to point towards a null value instead of the current tail



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

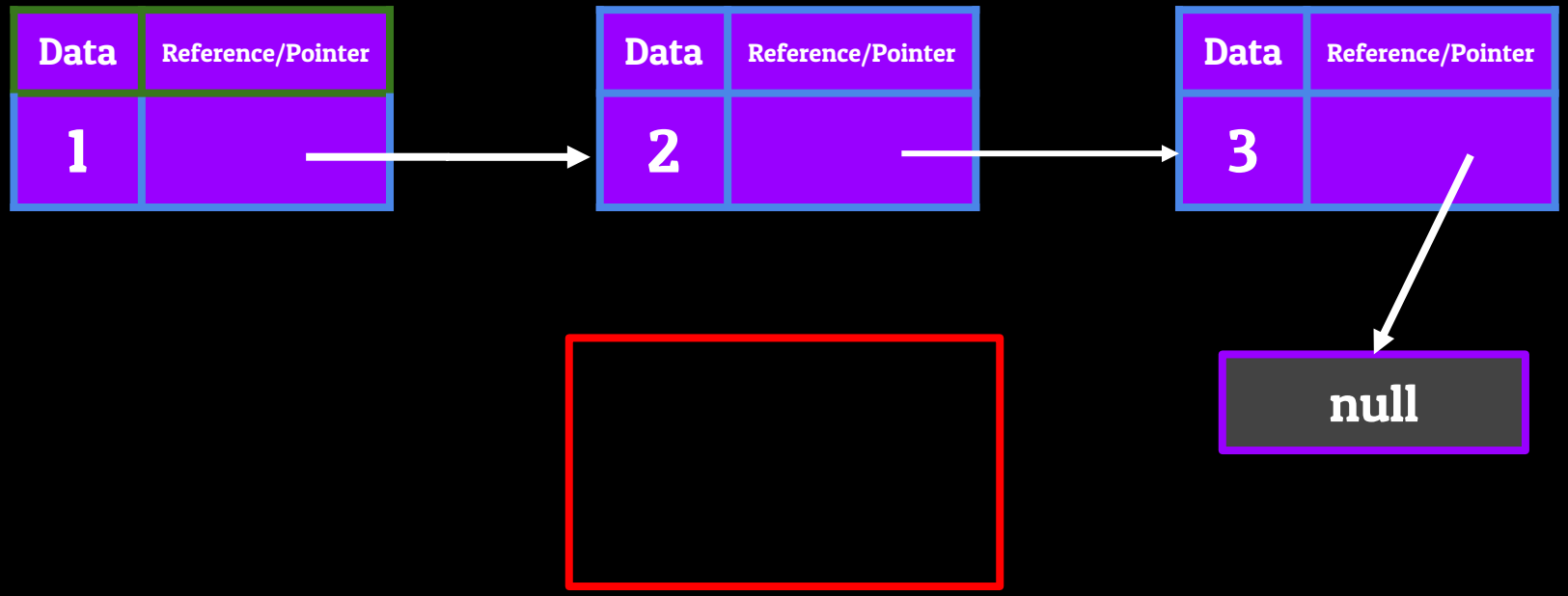
Set the previous tail to point towards a null value instead of the current tail



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

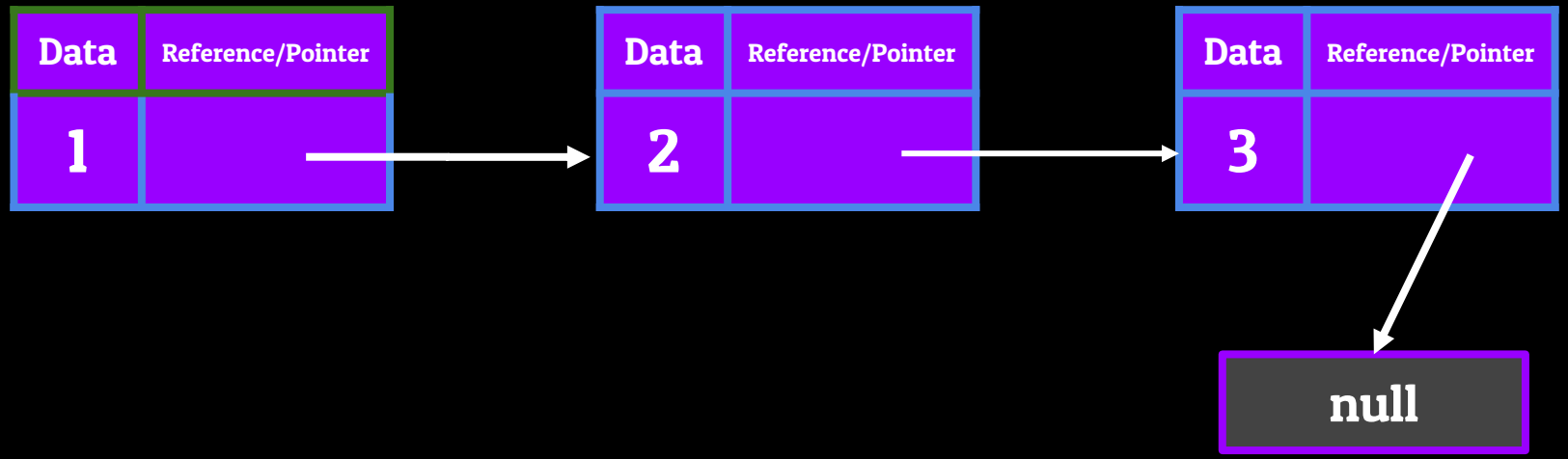
Set the previous tail to point towards a null value instead of the current tail



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

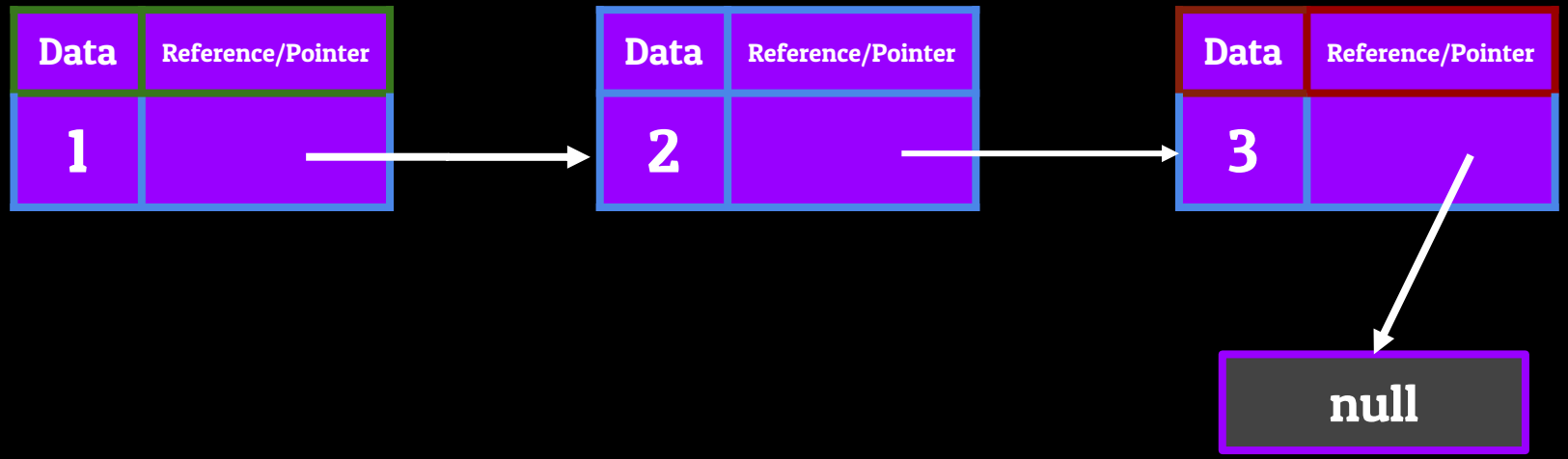
Set the previous tail to point towards a null value instead of the current tail



The Linked List - Adding and Removing Information

Removing from the Tail of a LinkedList

Set the previous tail to point towards a null value instead of the current tail



The LinkedList - Time Complexity Equations

The LinkedList - Time Complexity Equations

Accessing

The LinkedList - Time Complexity Equations

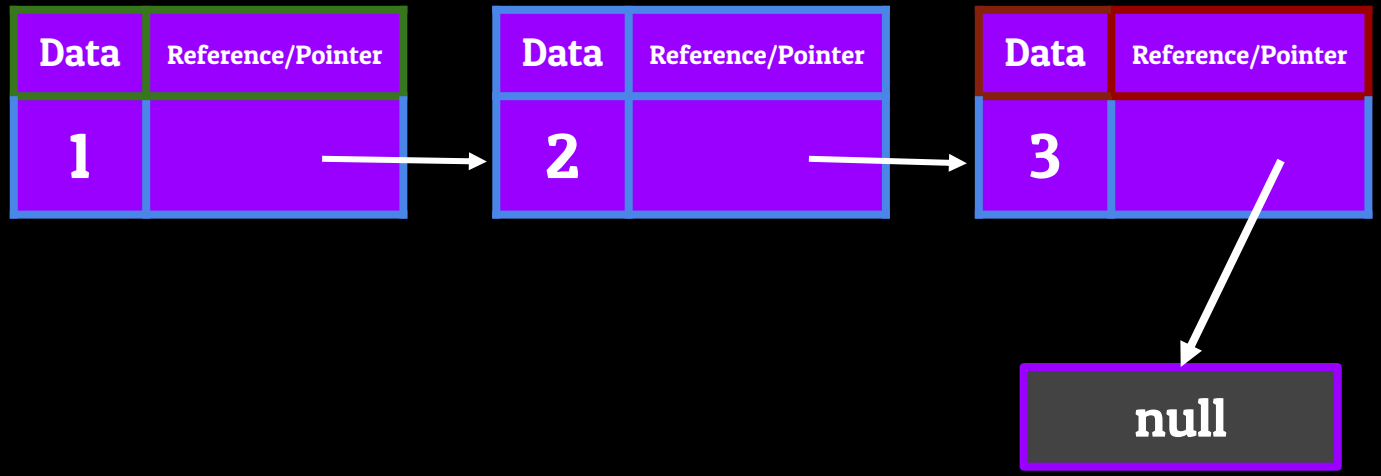
Accessing

O(n)

The LinkedList - Time Complexity Equations

Accessing

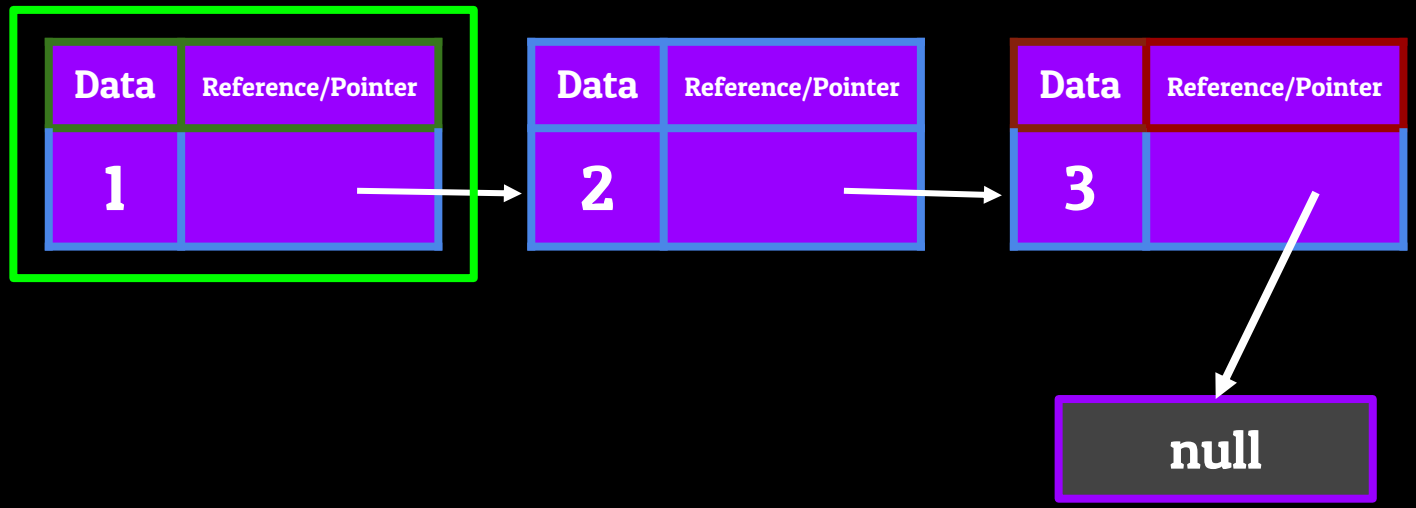
O(n)



The LinkedList - Time Complexity Equations

Accessing

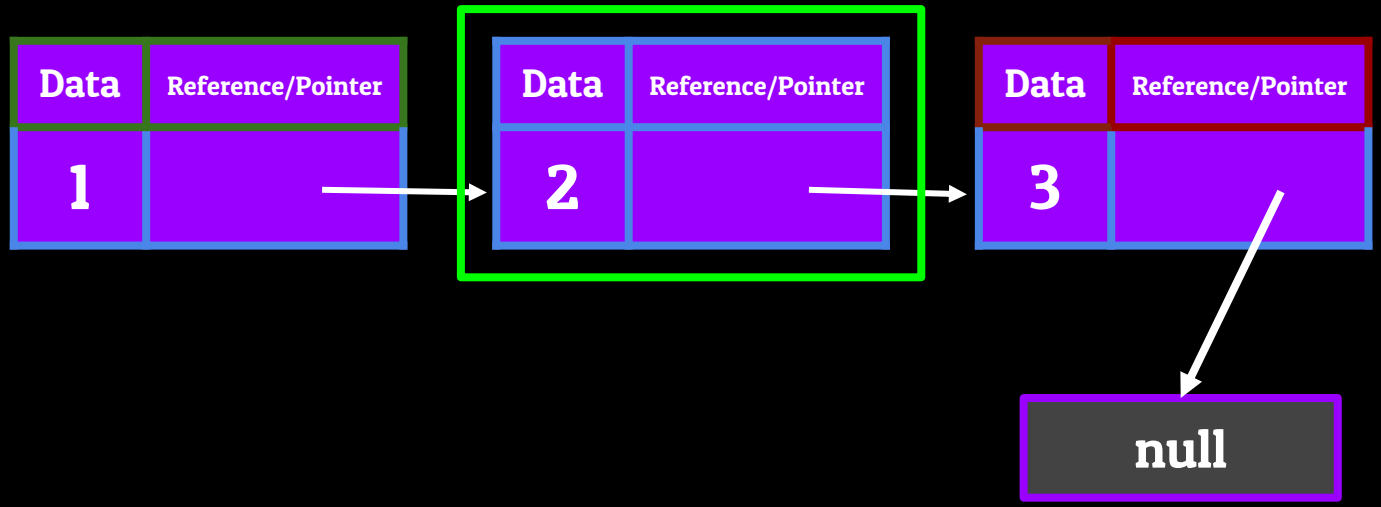
O(n)



The LinkedList - Time Complexity Equations

Accessing

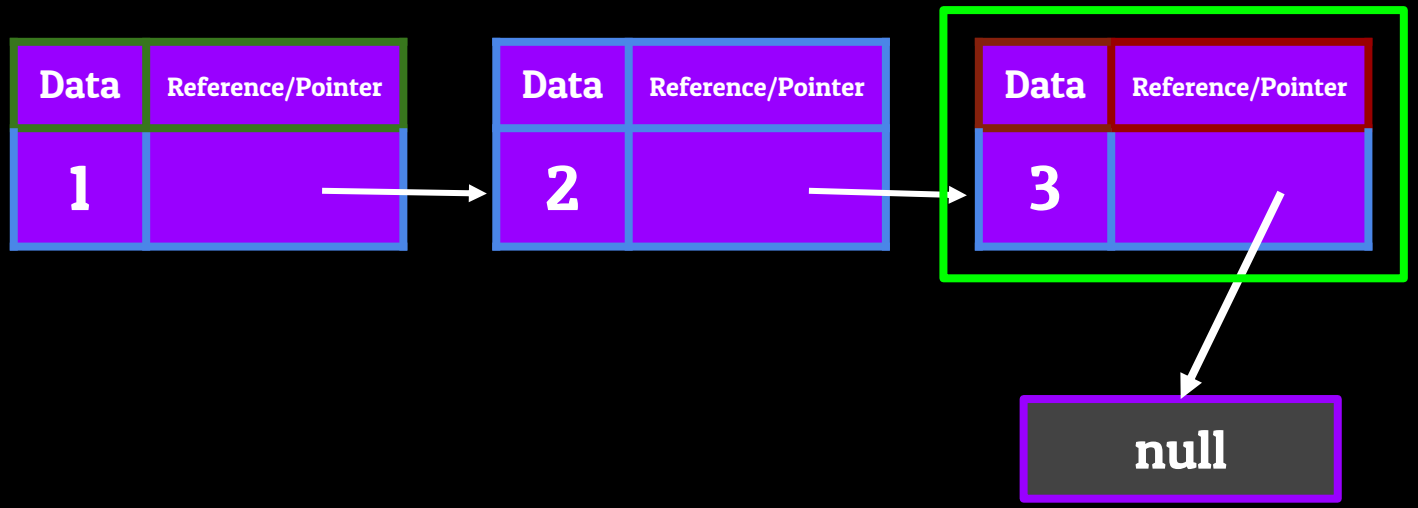
O(n)



The LinkedList - Time Complexity Equations

Accessing

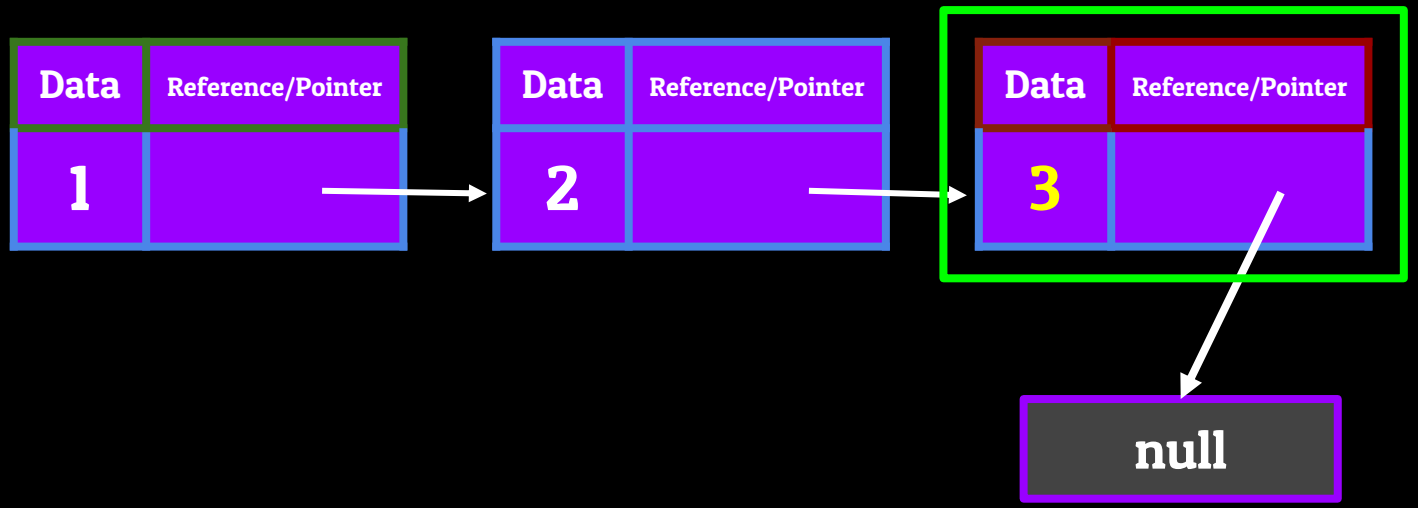
O(n)



The LinkedList - Time Complexity Equations

Accessing

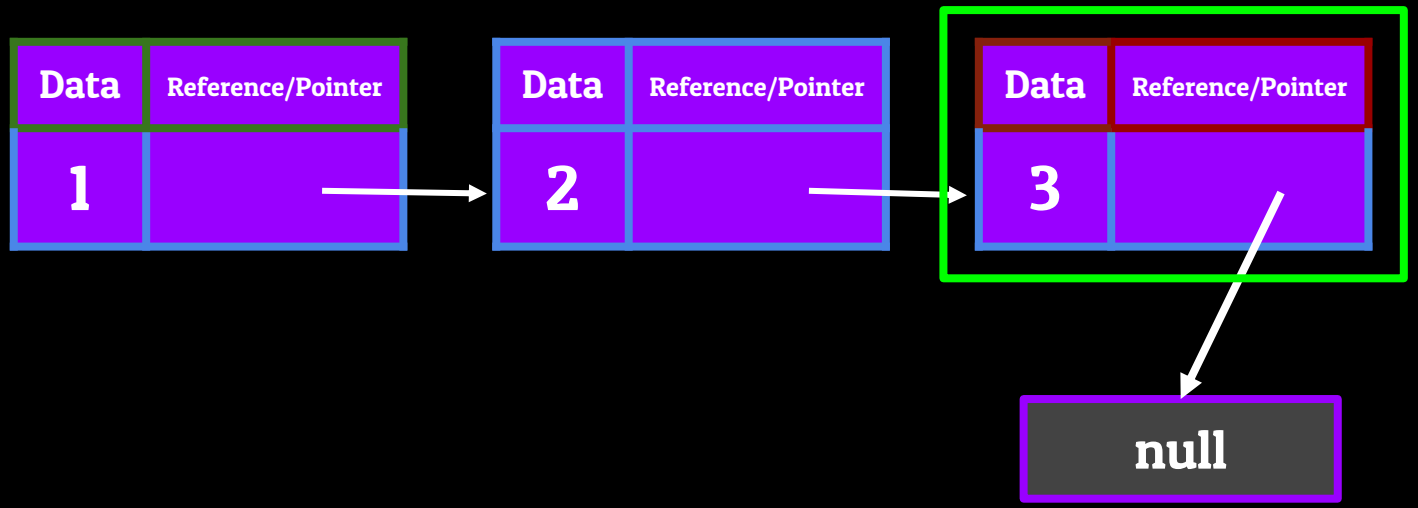
O(n)



The LinkedList - Time Complexity Equations

Accessing

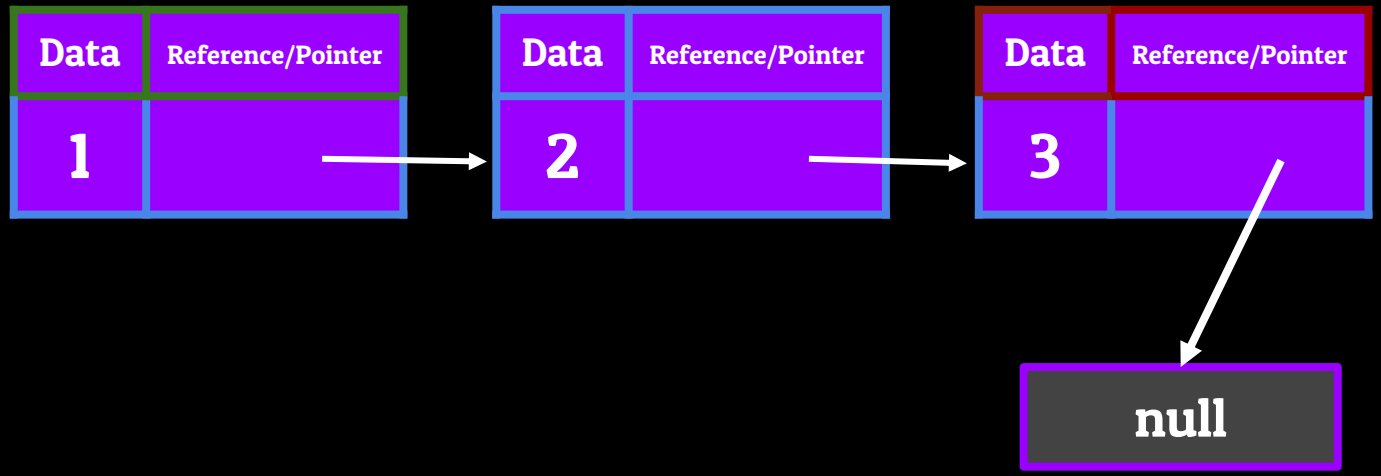
O(n)



The LinkedList - Time Complexity Equations

Accessing

O(n)



The LinkedList - Time Complexity Equations

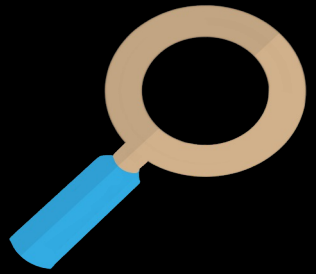
Accessing

O(n)

The LinkedList - Time Complexity Equations

Accessing

O(n)



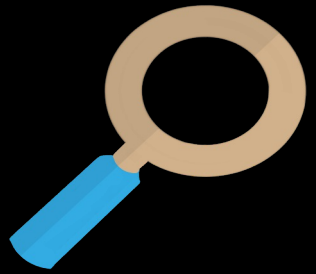
Searching

O(n)

The LinkedList - Time Complexity Equations

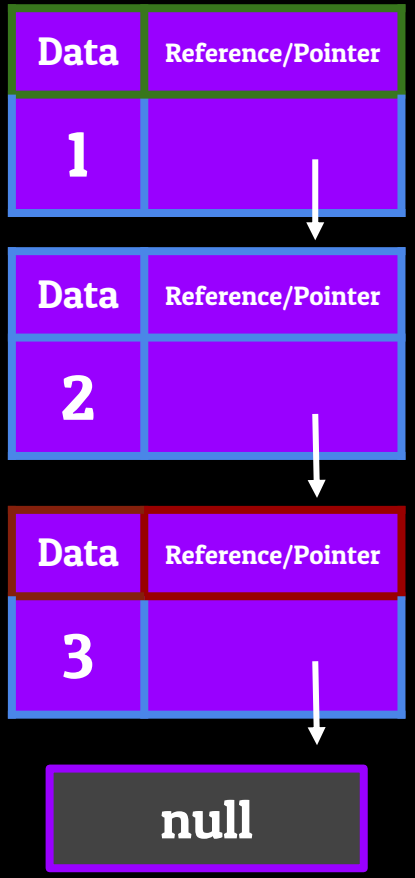
Accessing

O(n)



Searching

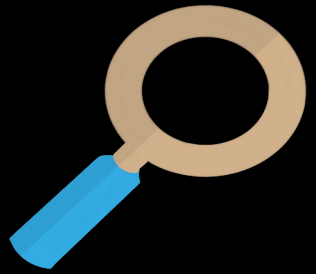
O(n)



The LinkedList - Time Complexity Equations

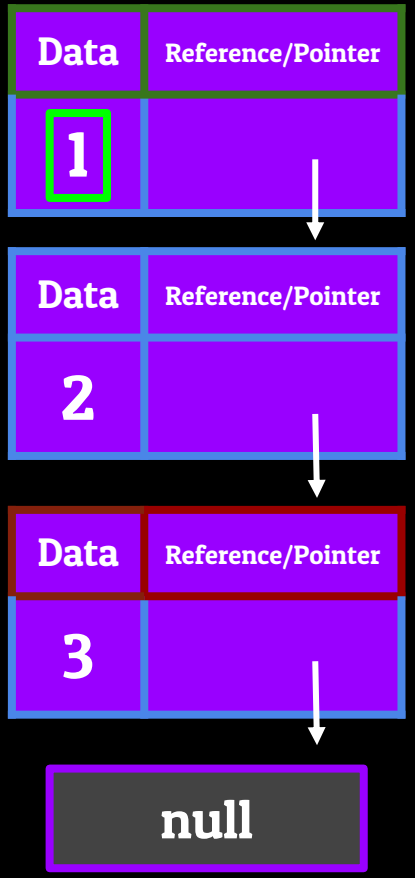
Accessing

O(n)



Searching

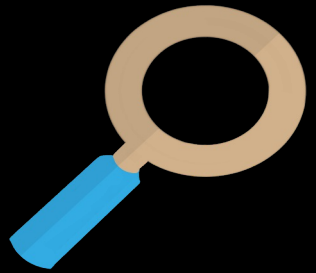
O(n)



The LinkedList - Time Complexity Equations

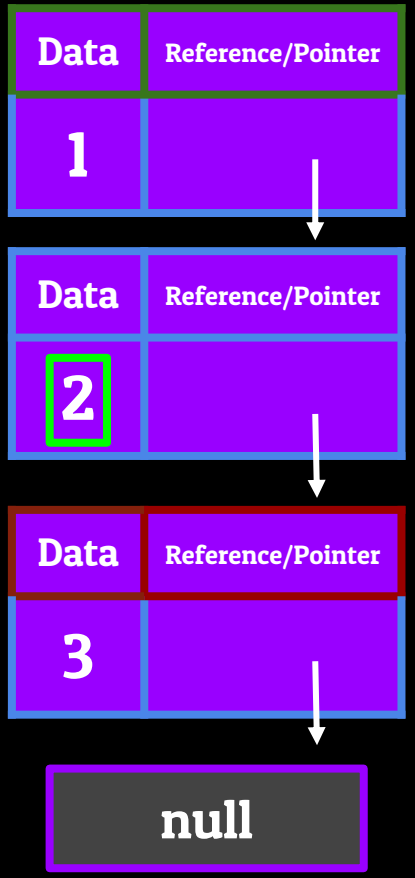
Accessing

O(n)



Searching

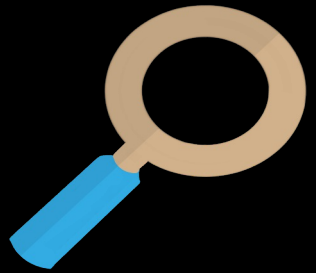
O(n)



The LinkedList - Time Complexity Equations

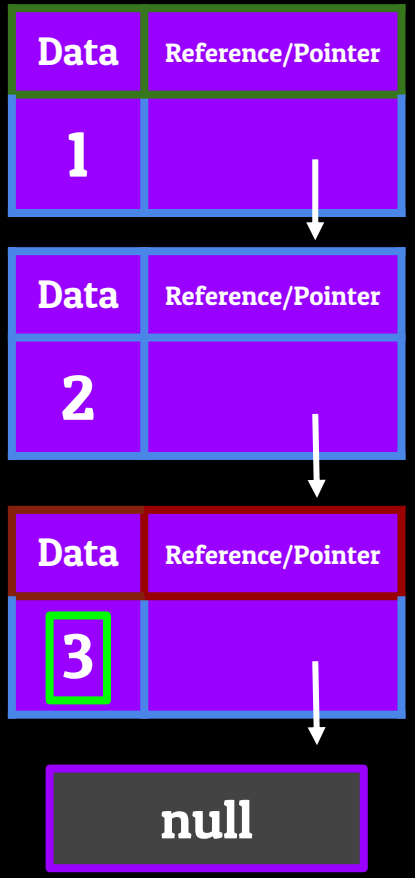
Accessing

O(n)



Searching

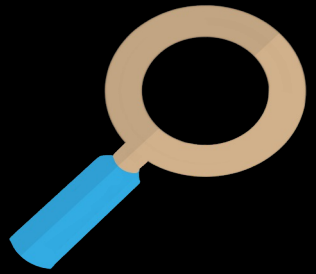
O(n)



The LinkedList - Time Complexity Equations

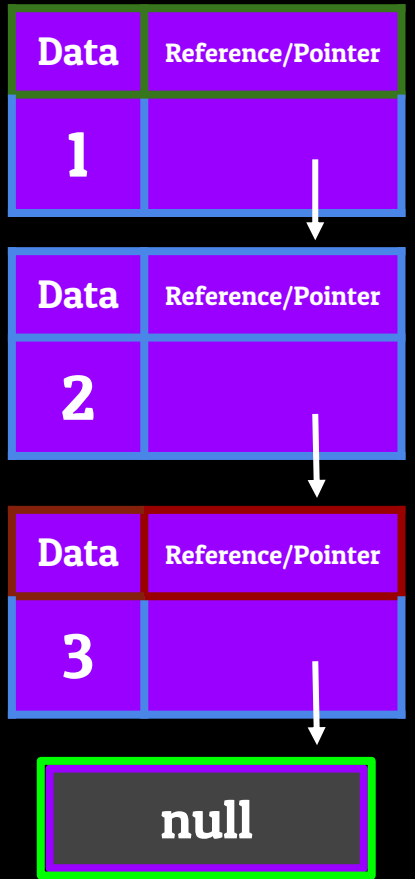
Accessing

O(n)



Searching

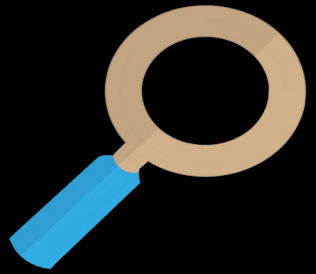
O(n)



The LinkedList - Time Complexity Equations

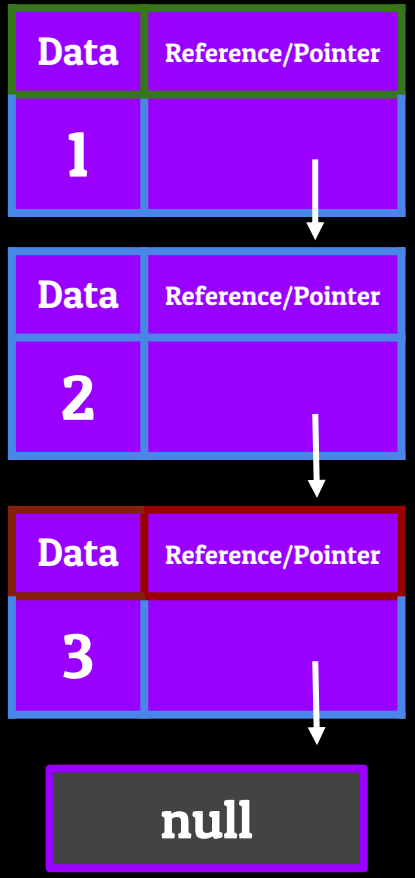
Accessing

O(n)



Searching

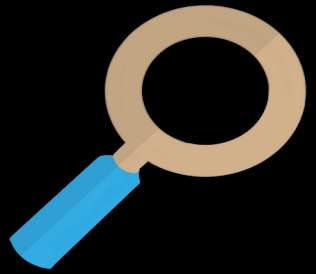
O(n)



The LinkedList - Time Complexity Equations

Accessing

O(n)



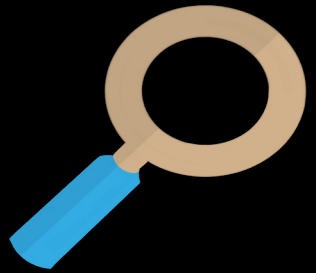
Searching

O(n)

The LinkedList - Time Complexity Equations

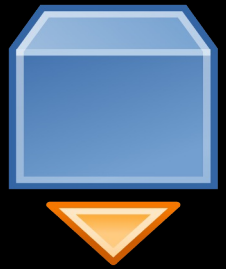
Accessing

O(n)



Searching

O(n)



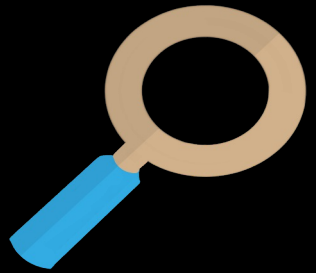
Inserting

Deleting

The LinkedList - Time Complexity Equations

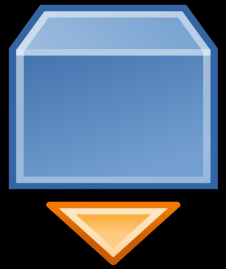
Accessing

O(n)



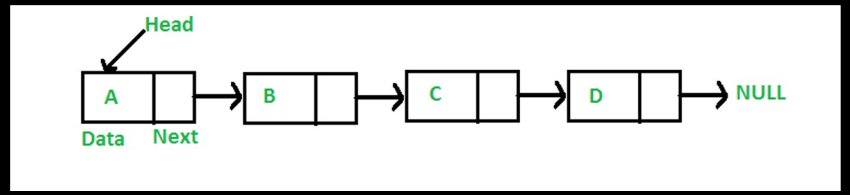
Searching

O(n)



Inserting

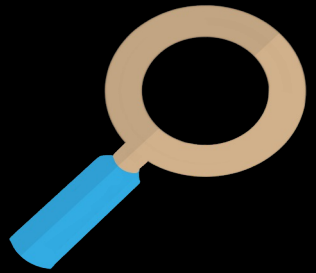
Deleting



The LinkedList - Time Complexity Equations

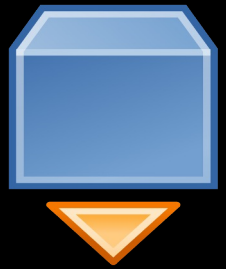
Accessing

O(n)



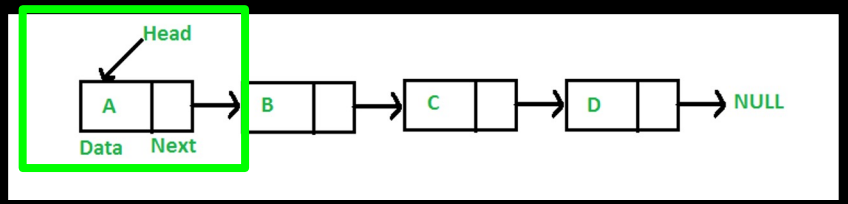
Searching

O(n)



Inserting

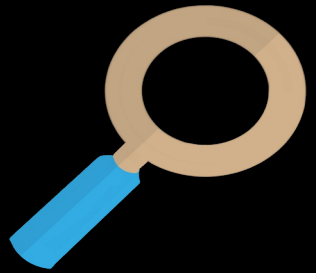
Deleting



The LinkedList - Time Complexity Equations

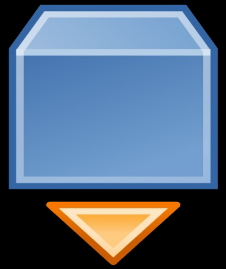
Accessing

O(n)



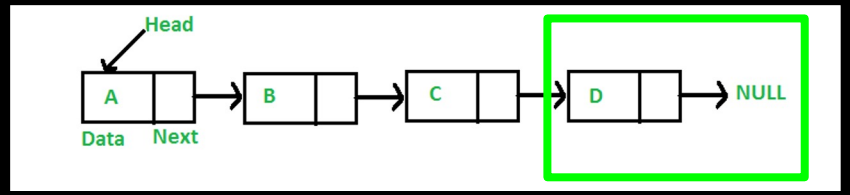
Searching

O(n)



Inserting

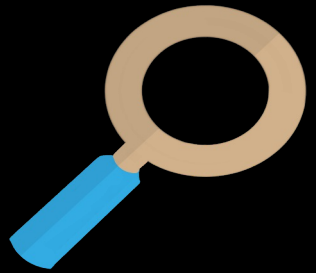
Deleting



The LinkedList - Time Complexity Equations

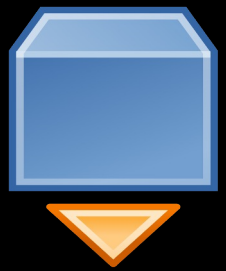
Accessing

O(n)



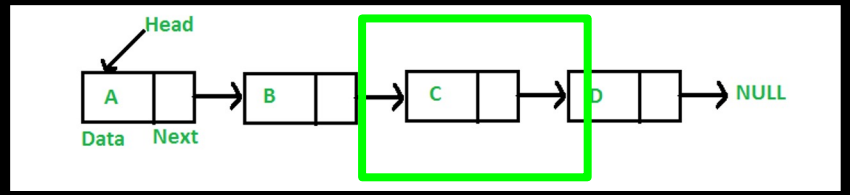
Searching

O(n)



Inserting

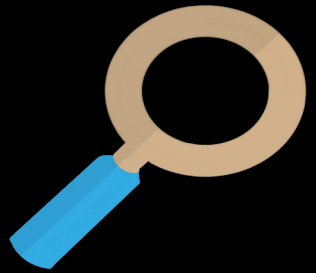
Deleting



The LinkedList - Time Complexity Equations

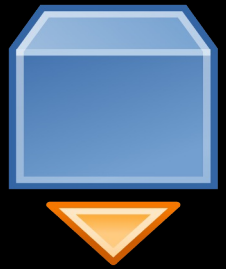
Accessing

O(n)



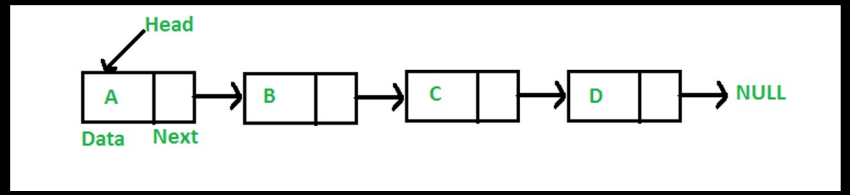
Searching

O(n)



Inserting

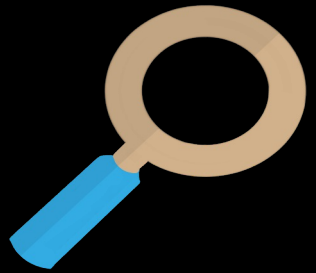
Deleting



The LinkedList - Time Complexity Equations

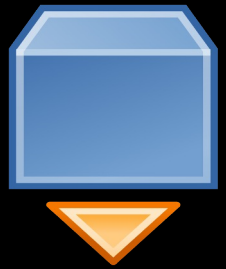
Accessing

O(n)



Searching

O(n)



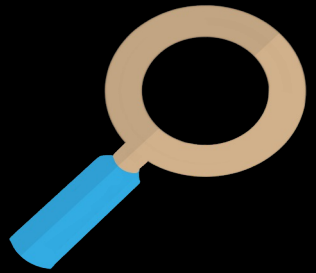
Inserting

Deleting

The LinkedList - Time Complexity Equations

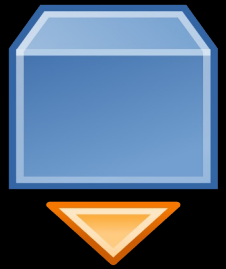
Accessing

O(n)



Searching

O(n)



Inserting

O(n)

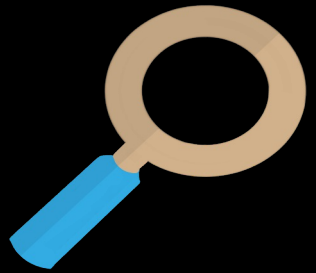
Deleting

O(n)

The LinkedList - Time Complexity Equations

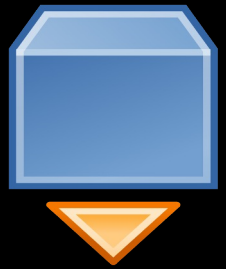
Accessing

O(n)



Searching

O(n)



Inserting

O(n) O(1)

Deleting

O(n) O(1)

The LinkedList - Uses for LinkedLists

The LinkedList - Uses for LinkedLists

- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.

The LinkedList - Uses for LinkedLists

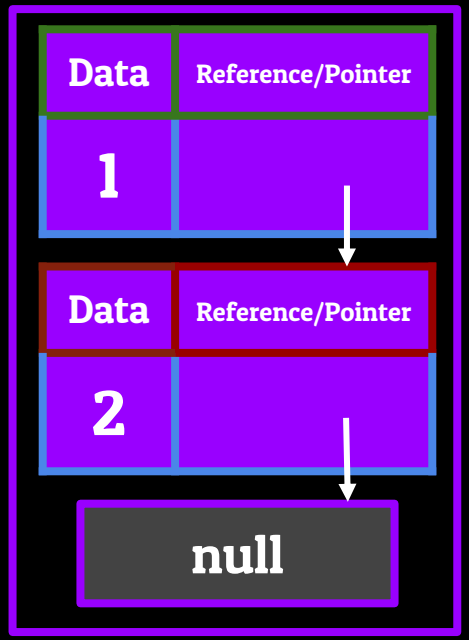
- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.

The LinkedList - Uses for LinkedLists

- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.

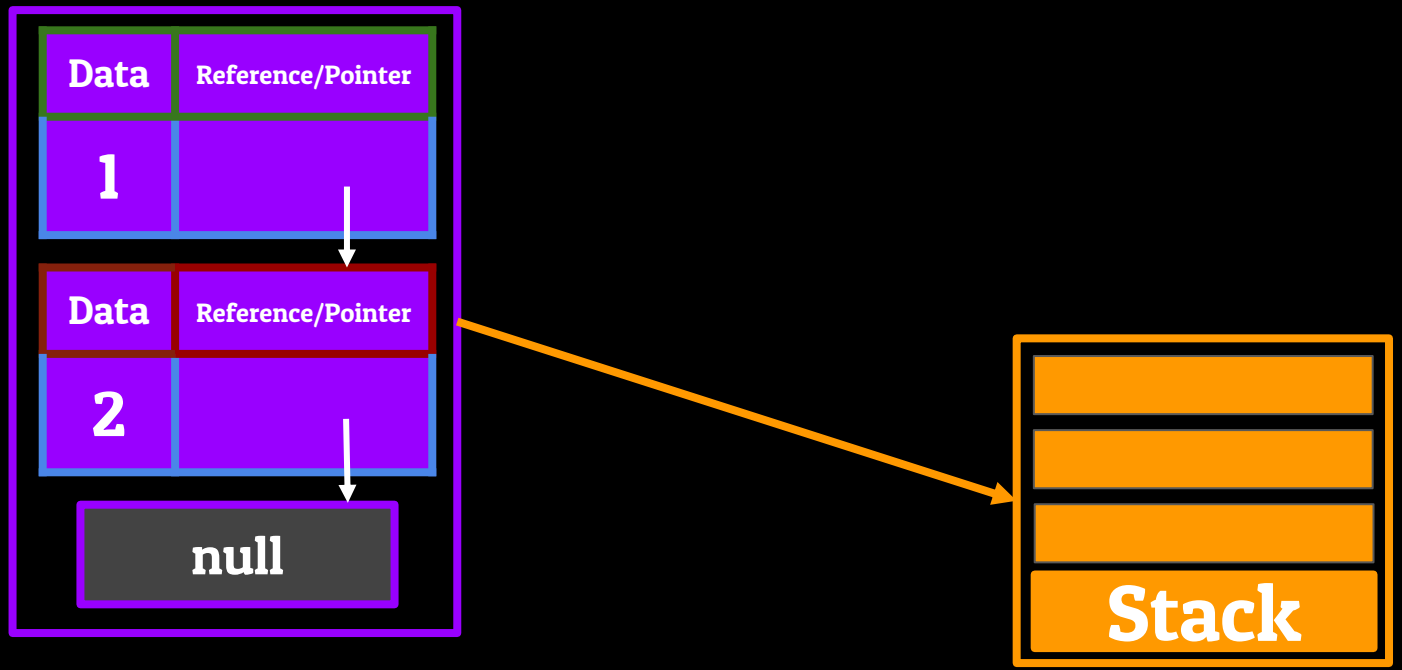
The LinkedList - Uses for LinkedLists

- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.



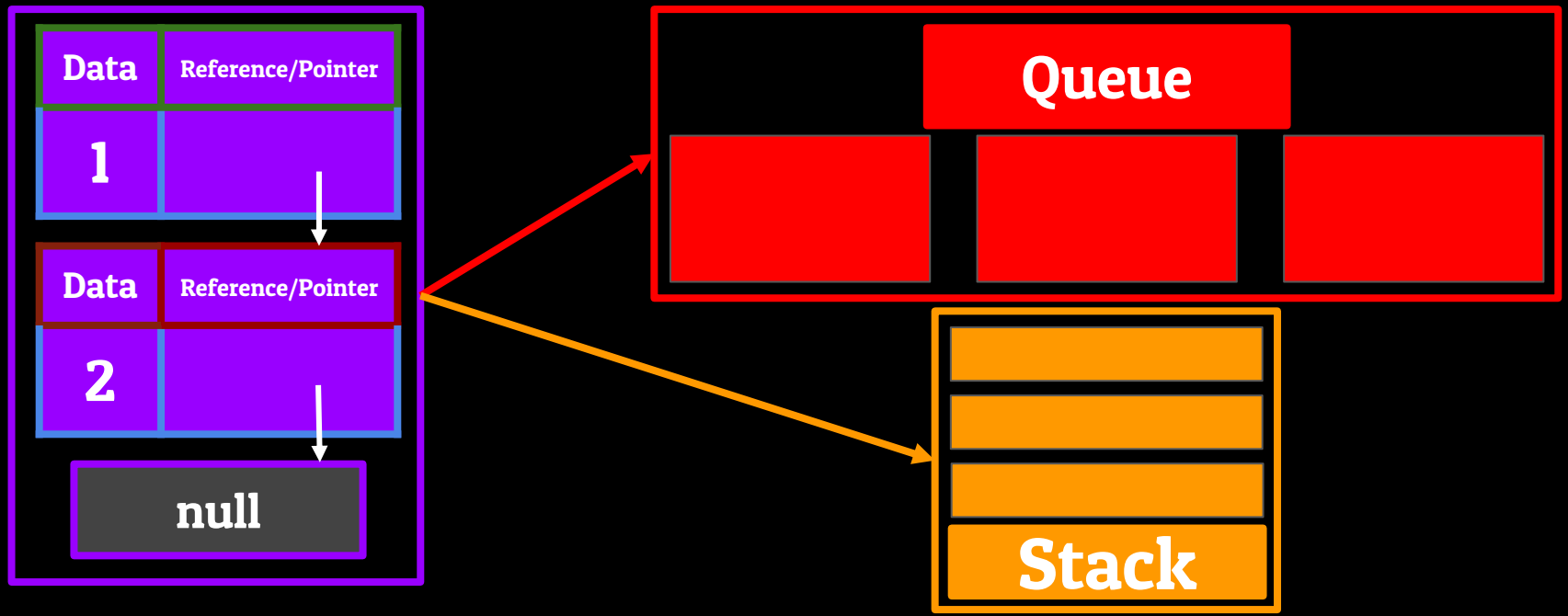
The LinkedList - Uses for LinkedLists

- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.



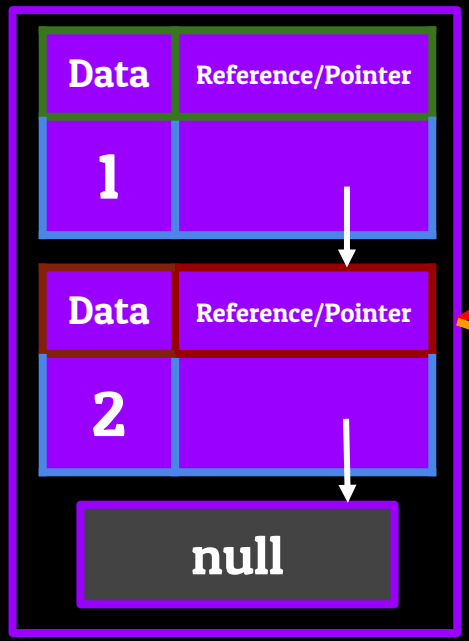
The LinkedList - Uses for LinkedLists

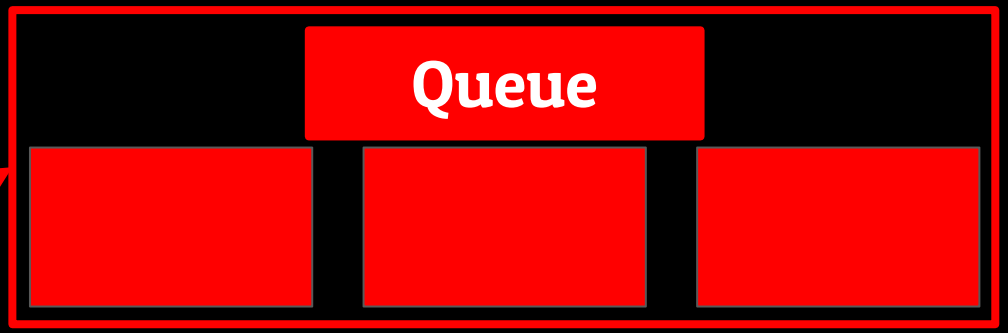
- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.

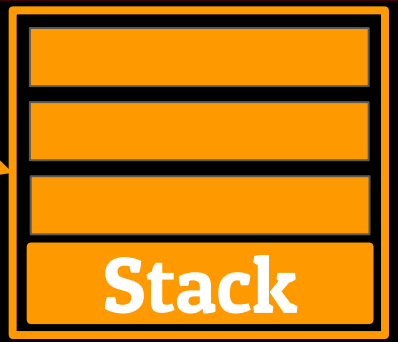


The LinkedList - Uses for LinkedLists

- LinkedLists can be used in the backing of other data structures
  - We can use LinkedLists to make Stacks, Queues, etc.







Article in The Description

The LinkedList - Uses for LinkedLists

The LinkedList - Uses for LinkedLists

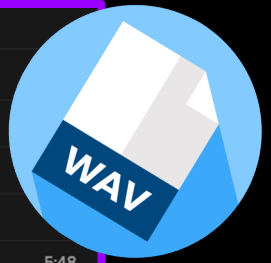
NEXT TRACKS

#		SONG	ARTIST	ALBUM	
1	+	Don't Believe Her	Scorpions	Crazy World	4:56
2	+	To Be With You In Heaven	Scorpions	Crazy World	4:52
3	+	Wind Of Change	Scorpions	Crazy World	5:13
4	+	Restless Nights	Scorpions	Crazy World	5:48
5	+	Lust Or Love	Scorpions	Crazy World	4:23
6	+	Kicks After Six	Scorpions	Crazy World	3:50
7	+	Hit Between The Eyes	Scorpions	Crazy World	4:34
8	+	Money And Fame	Scorpions	Crazy World	5:07
9	+	Crazy World	Scorpions	Crazy World	5:09
10	+	Send Me An Angel	Scorpions	Crazy World	4:33

The LinkedList - Uses for LinkedLists

NEXT TRACKS

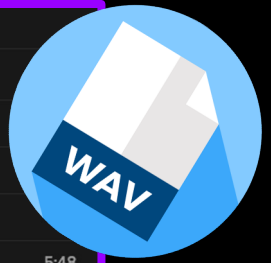
#		SONG	ARTIST	ALBUM	
1	+	Don't Believe Her	Scorpions	Crazy World	
2	+	To Be With You In Heaven	Scorpions	Crazy World	
3	+	Wind Of Change	Scorpions	Crazy World	
4	+	Restless Nights	Scorpions	Crazy World	5:48
5	+	Lust Or Love	Scorpions	Crazy World	4:23
6	+	Kicks After Six	Scorpions	Crazy World	3:50
7	+	Hit Between The Eyes	Scorpions	Crazy World	4:34
8	+	Money And Fame	Scorpions	Crazy World	5:07
9	+	Crazy World	Scorpions	Crazy World	5:09
10	+	Send Me An Angel	Scorpions	Crazy World	4:33



The LinkedList - Uses for LinkedLists

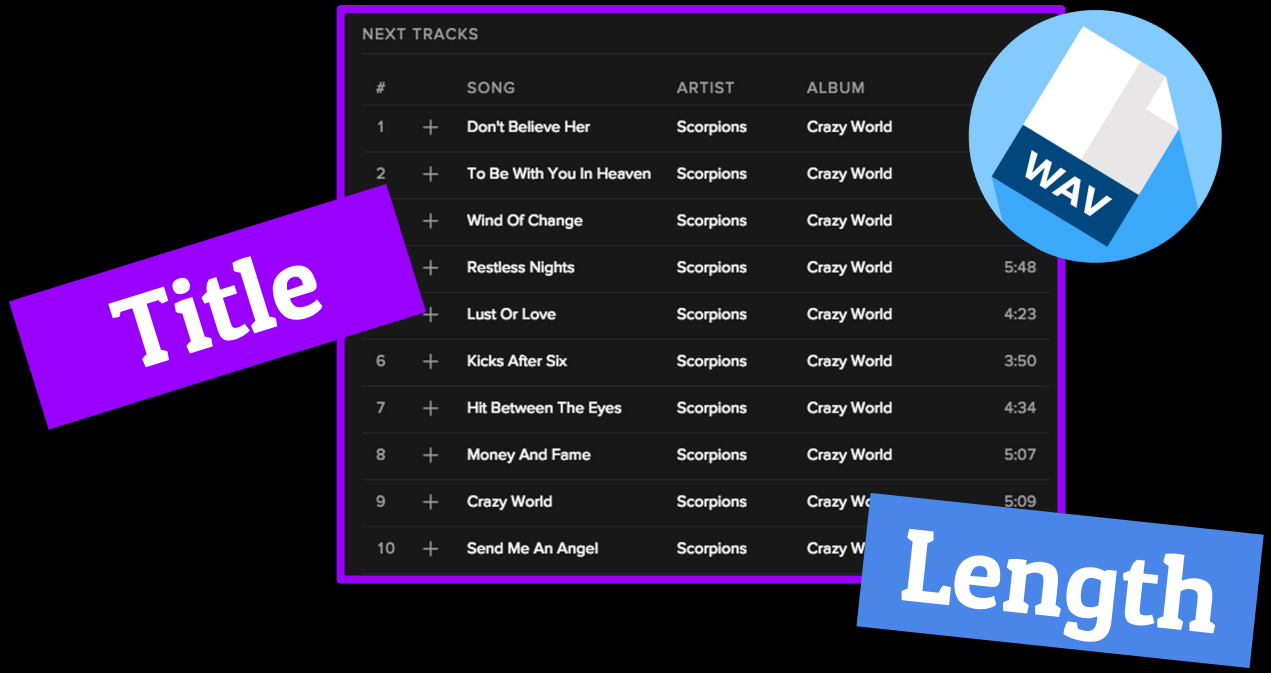
NEXT TRACKS

#		SONG	ARTIST	ALBUM	
1	+	Don't Believe Her	Scorpions	Crazy World	
2	+	To Be With You In Heaven	Scorpions	Crazy World	
	+	Wind Of Change	Scorpions	Crazy World	
	+	Restless Nights	Scorpions	Crazy World	5:48
	+	Lust Or Love	Scorpions	Crazy World	4:23
6	+	Kicks After Six	Scorpions	Crazy World	3:50
7	+	Hit Between The Eyes	Scorpions	Crazy World	4:34
8	+	Money And Fame	Scorpions	Crazy World	5:07
9	+	Crazy World	Scorpions	Crazy World	5:09
10	+	Send Me An Angel	Scorpions	Crazy World	4:33

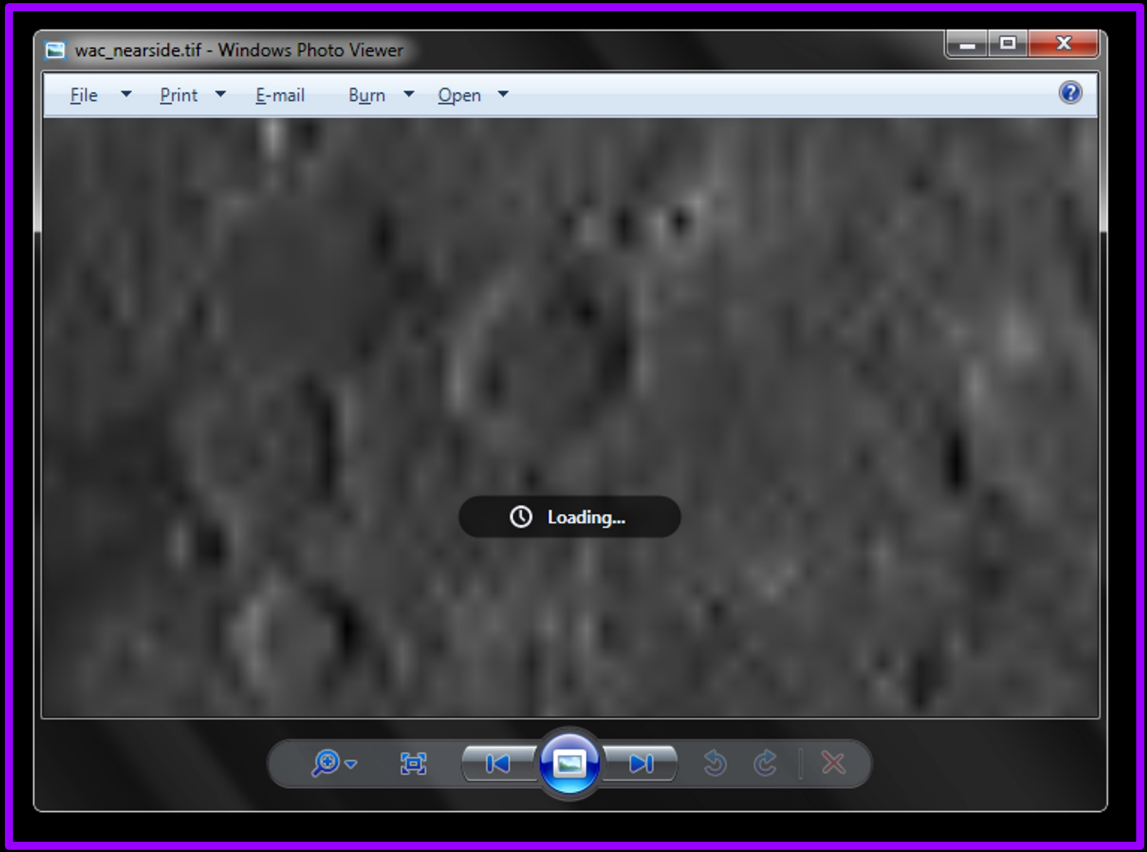


Title

The LinkedList - Uses for LinkedLists

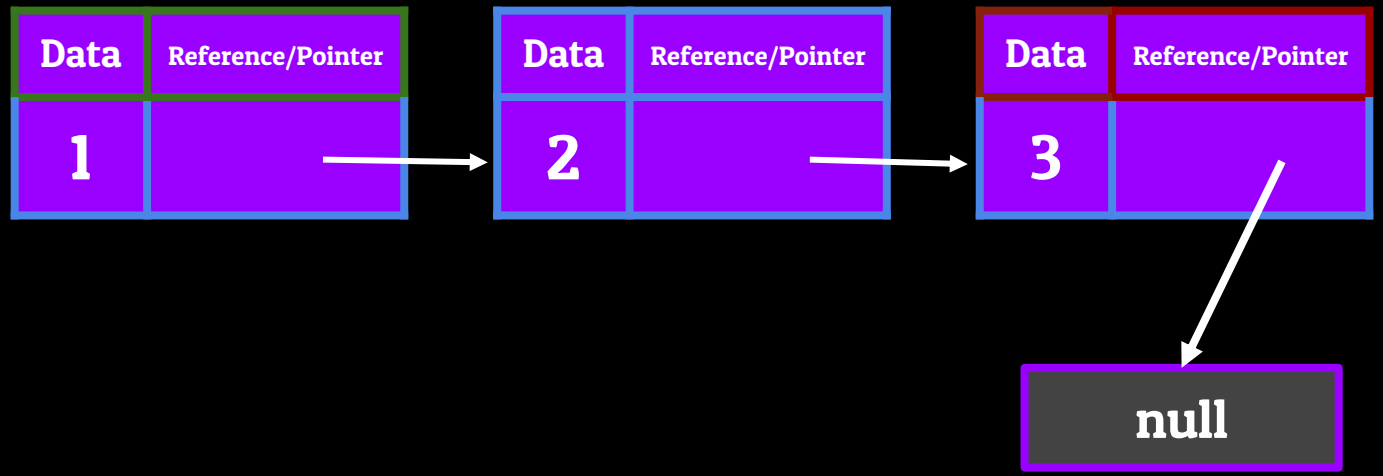


The LinkedList - Uses for LinkedLists



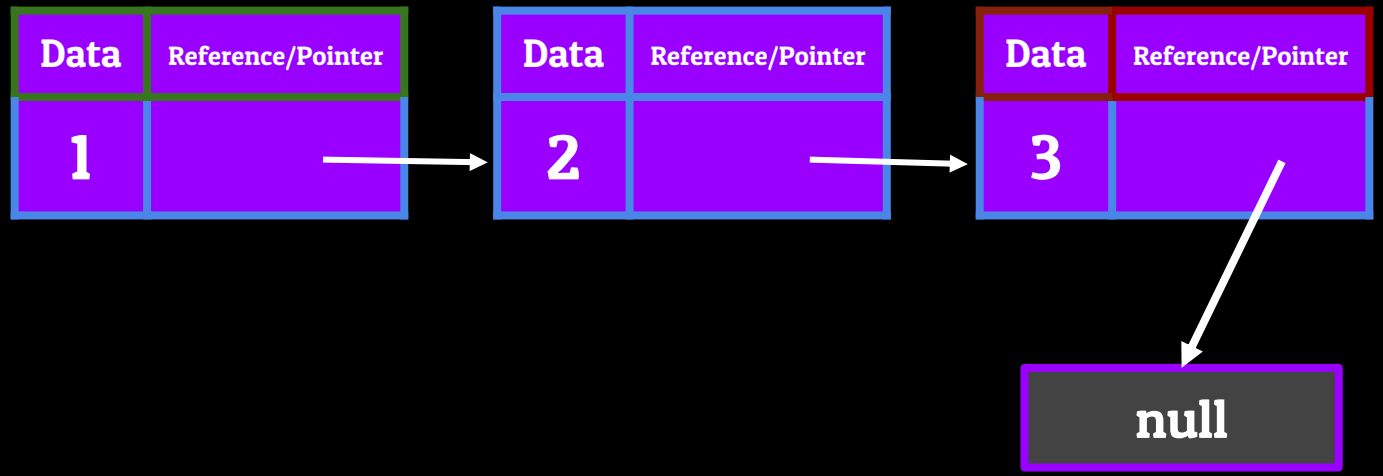
The LinkedList - Review + Conclusion

- A LinkedList is sequential access linear data structure in which every element is a separate object called a node, containing 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List

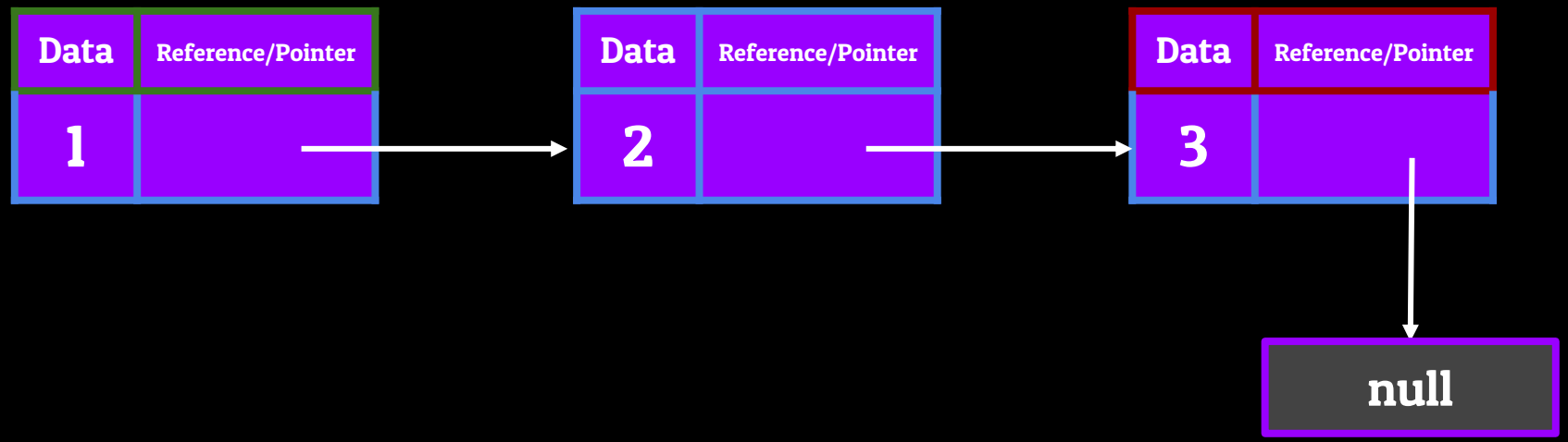


The LinkedList - Review + Conclusion

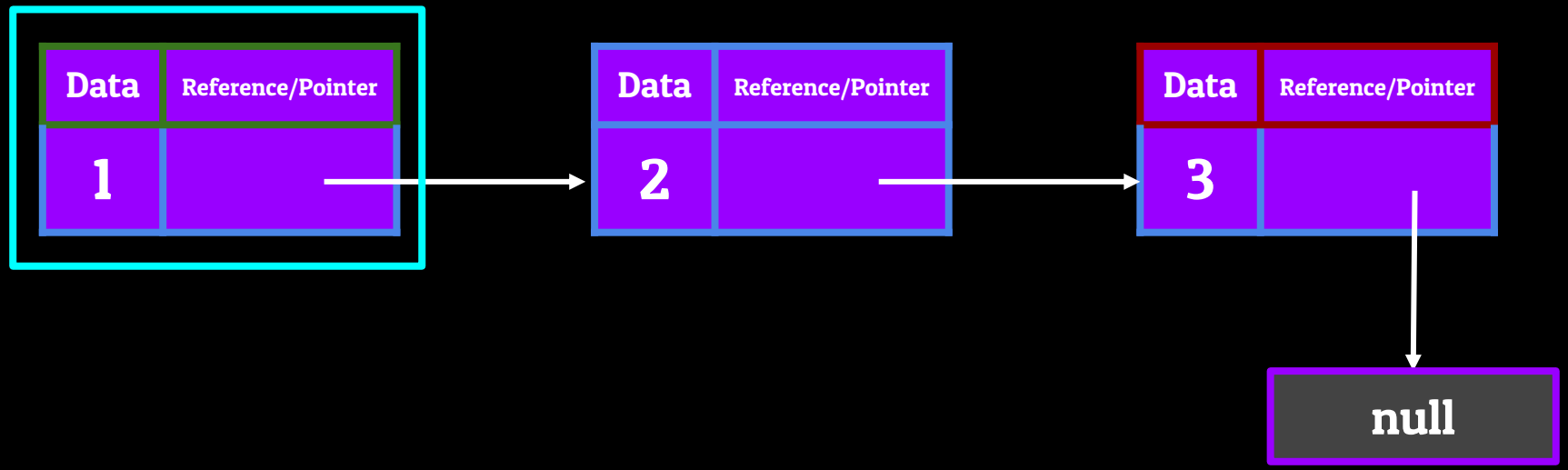
- A LinkedList is sequential access linear data structure in which every element is a separate object called a node, containing 2 parts
  - The data
  - The reference (or pointer) which points to the next Node in the List



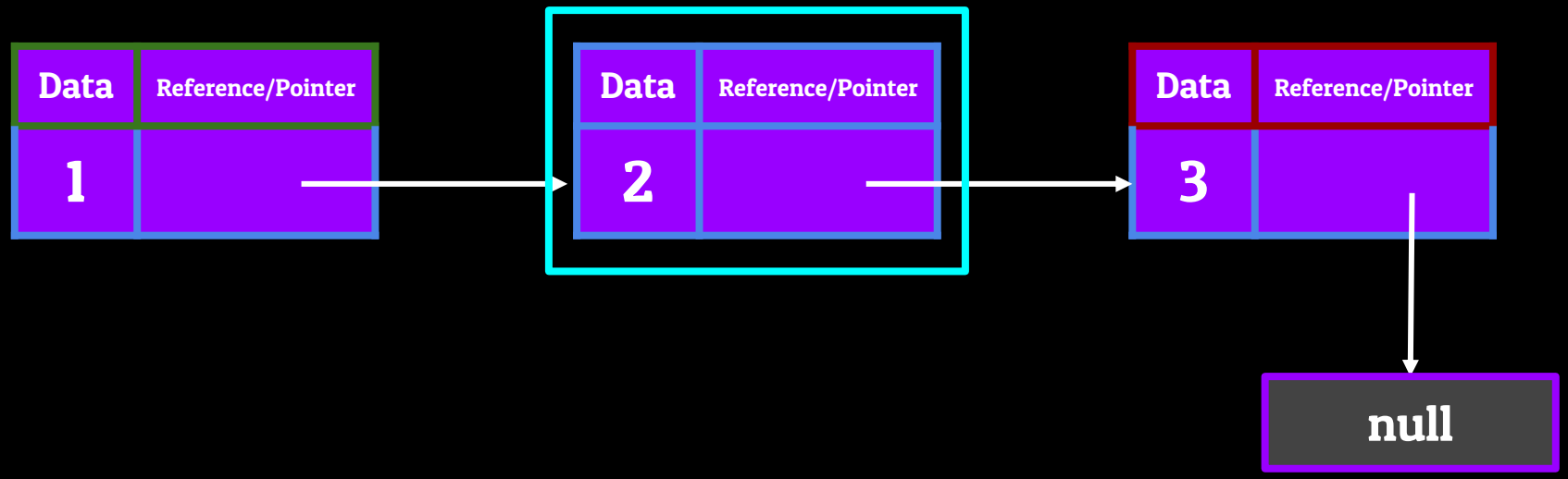
The LinkedList - One Big Drawback



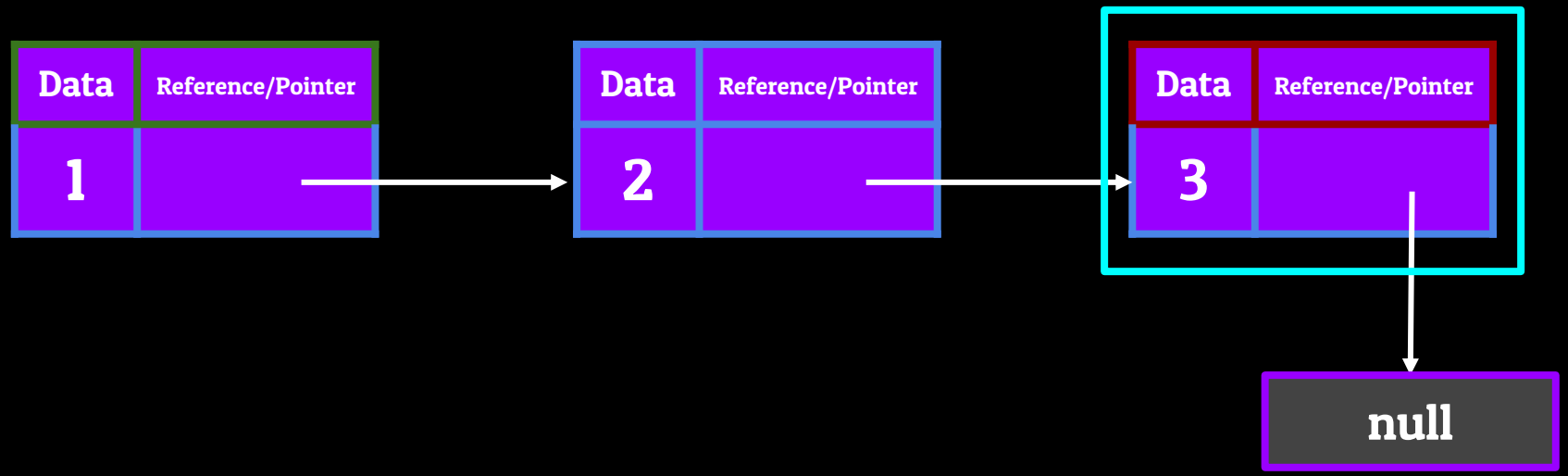
The LinkedList - One Big Drawback



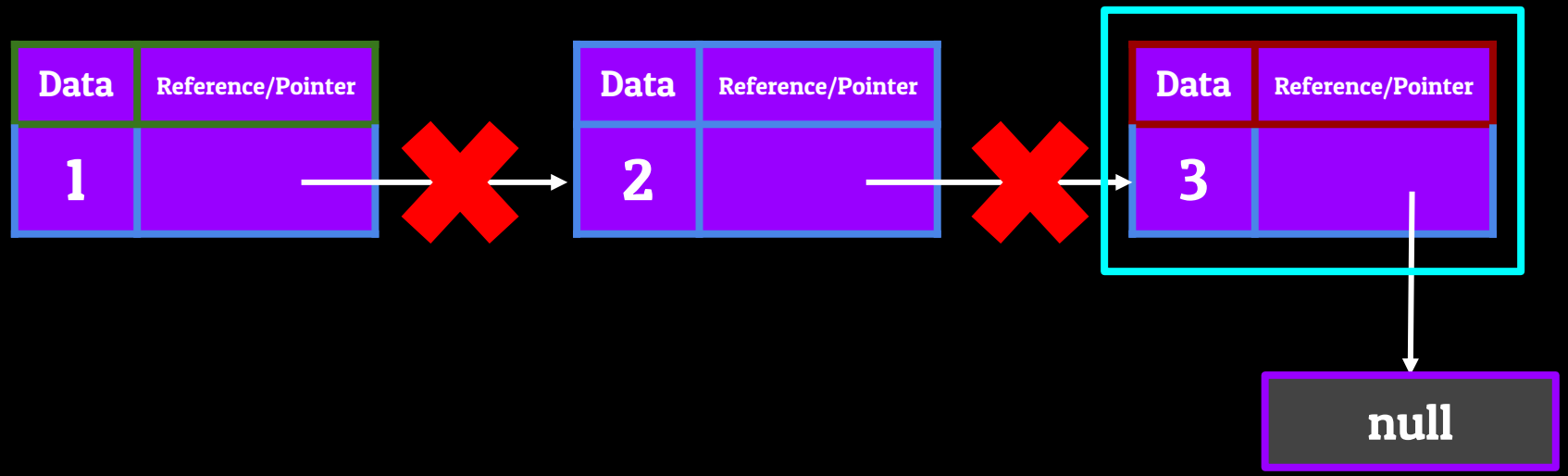
The LinkedList - One Big Drawback



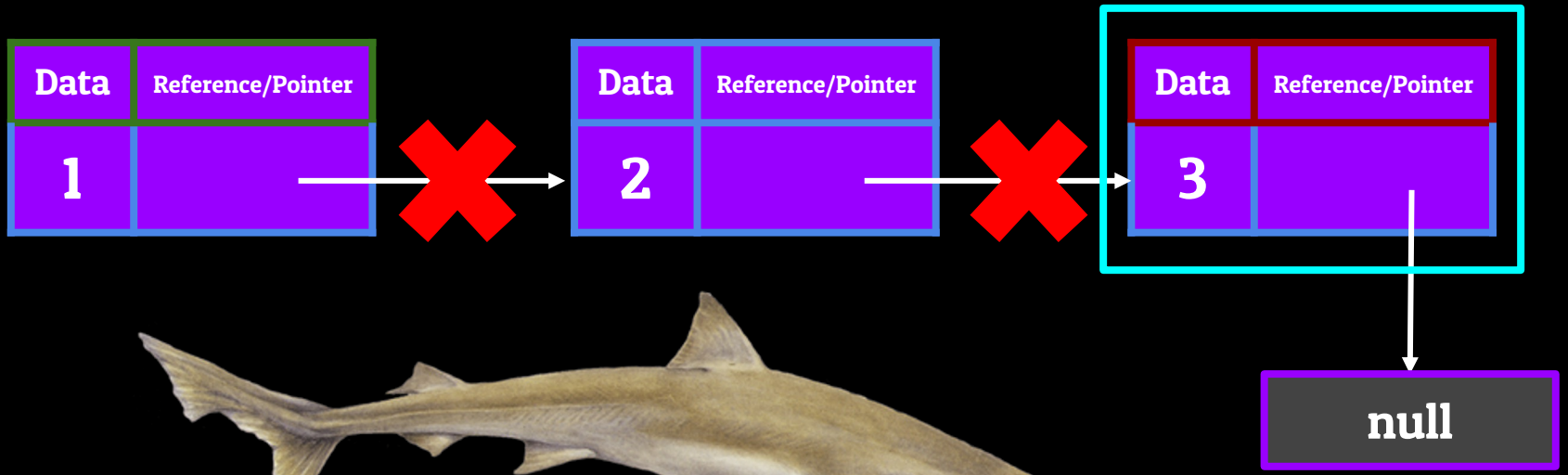
The LinkedList - One Big Drawback



The LinkedList - One Big Drawback



The LinkedList - One Big Drawback



An Introduction to Data Structures

The Doubly-LinkedList

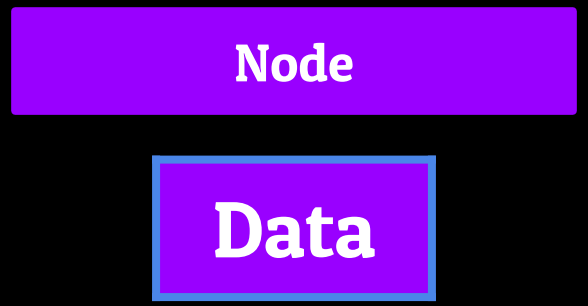
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers

Node

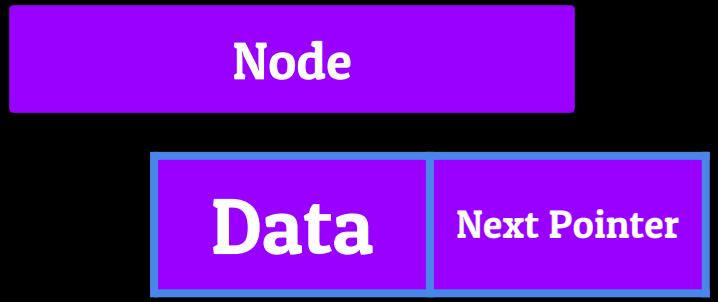
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



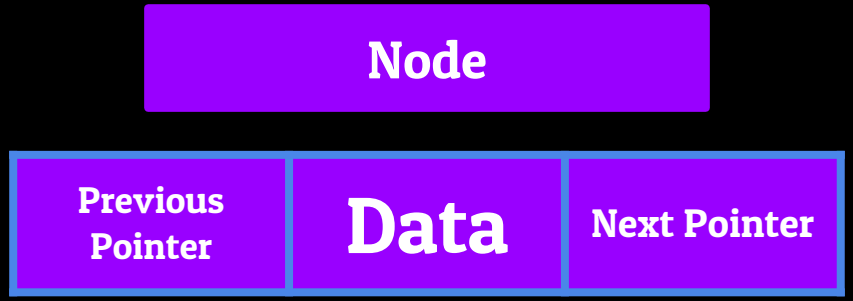
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



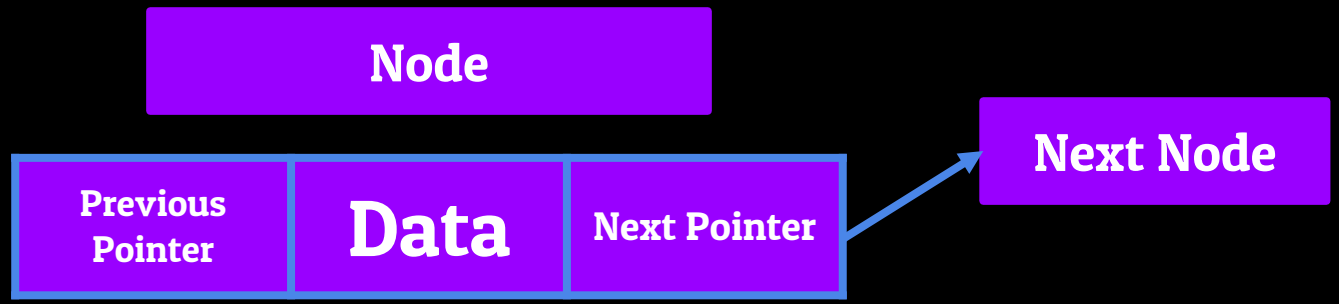
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



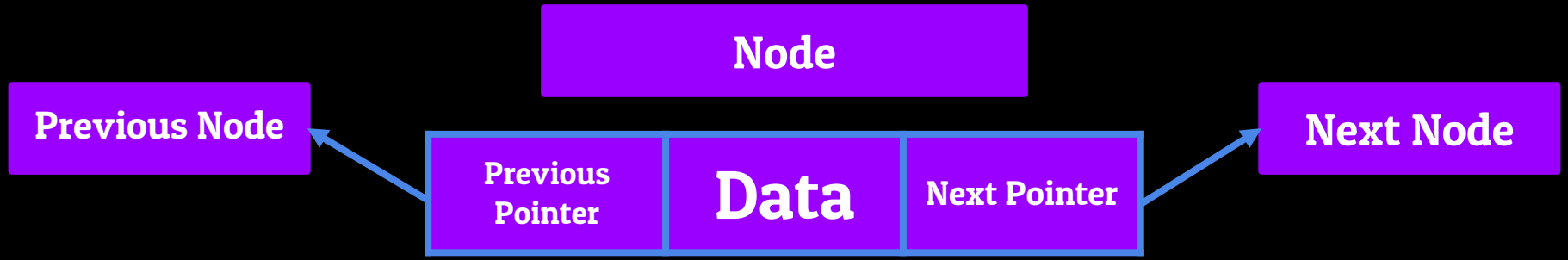
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



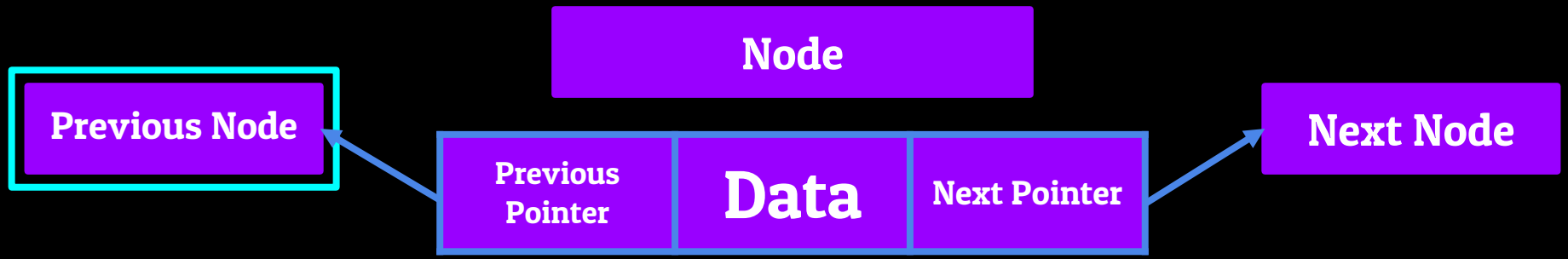
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



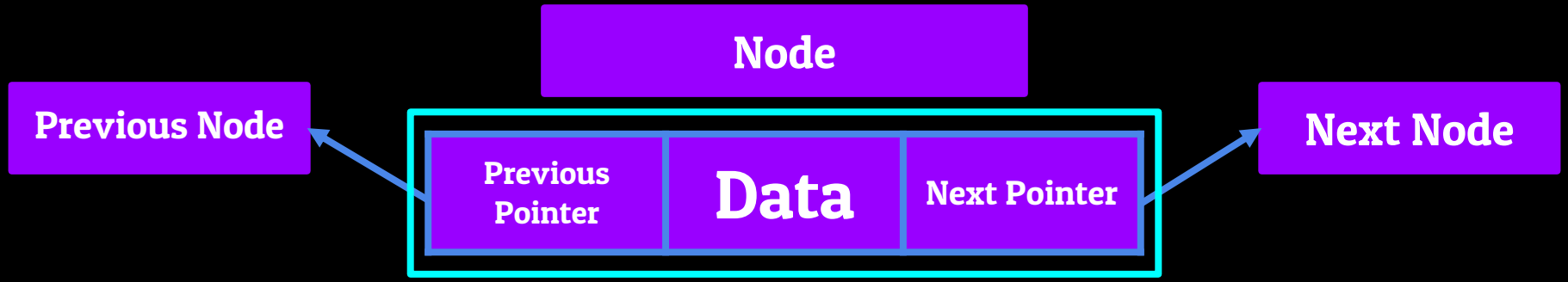
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



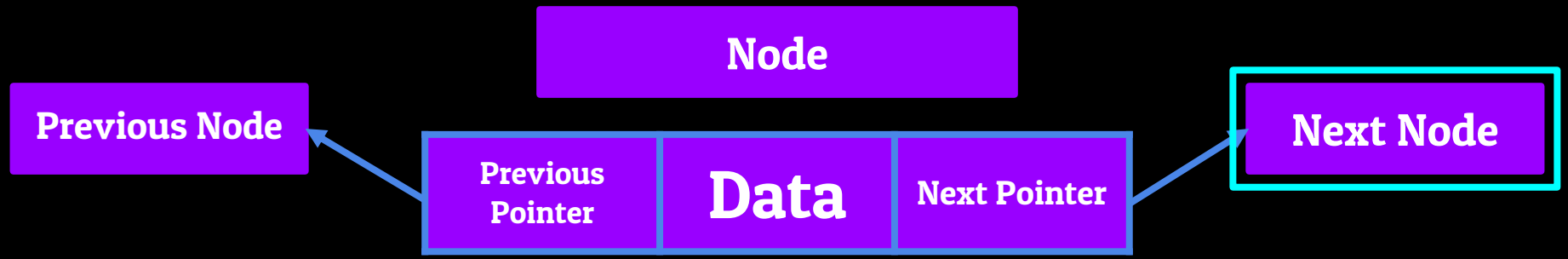
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



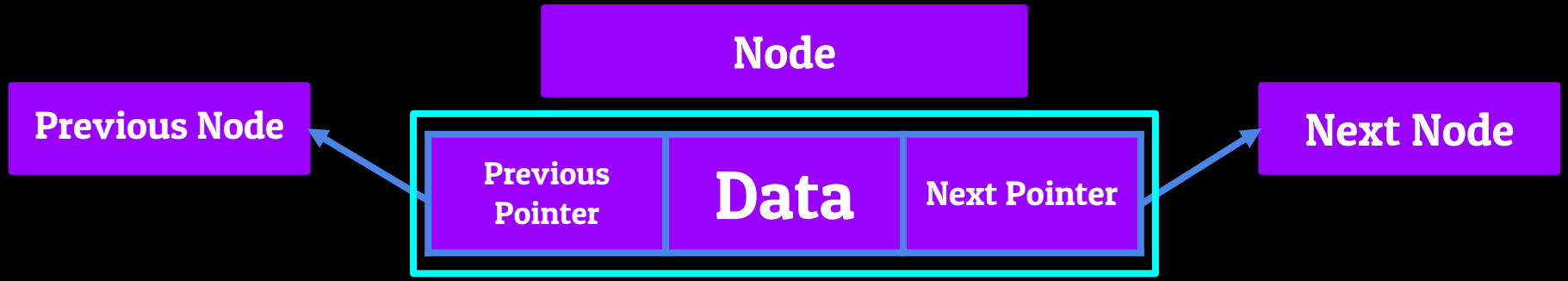
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



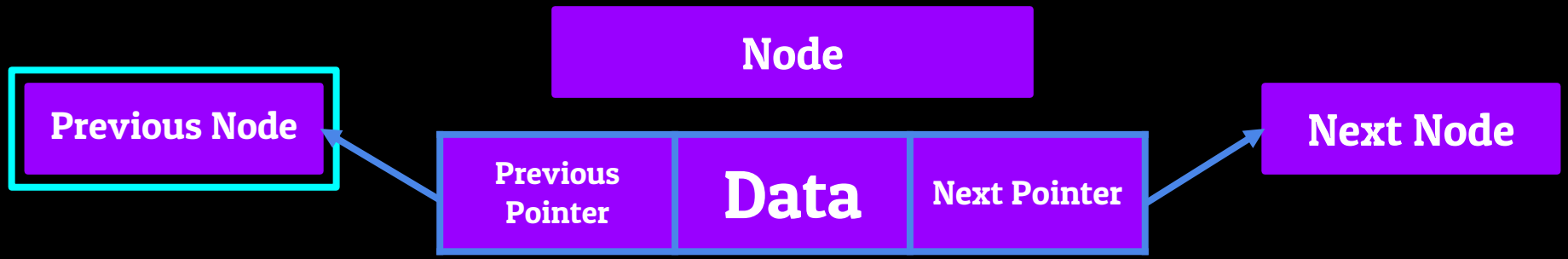
The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



The Doubly-Linked List - Background Information

- **The Doubly-Linked List** is a **sequential access data structure** which stores data in the form of Nodes
  - Able to traverse both **forwards** and **backwards** using pointers



The Doubly-Linked List - Visualization

"Next" = That particular Nodes pointer which points to the next object in the List

The Doubly-Linked List - Visualization

"Next" = That particular Nodes pointer which points to the next object in the List

"Previous" = That particular Nodes pointer which points to the previous object in the List

The Doubly-Linked List - Visualization

The Doubly-Linked List - Visualization

The Doubly-Linked List

The Doubly-Linked List - Visualization

The Doubly-Linked List

Head Node

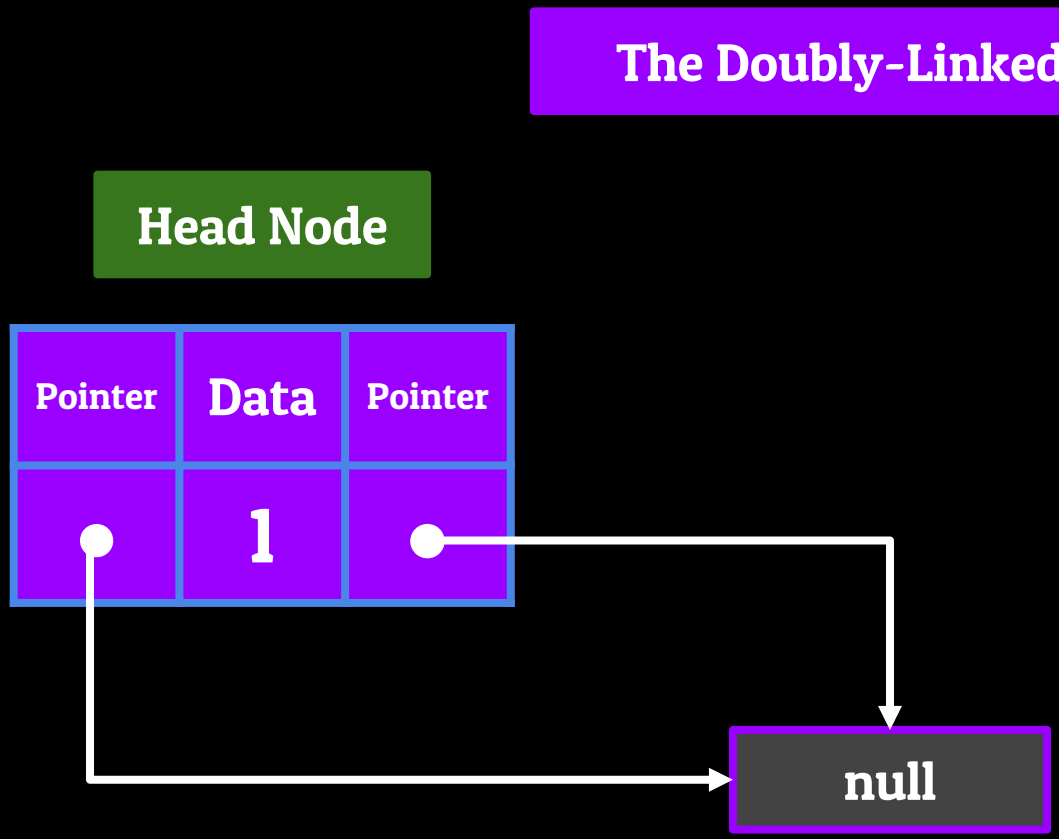
The Doubly-Linked List - Visualization

The Doubly-Linked List

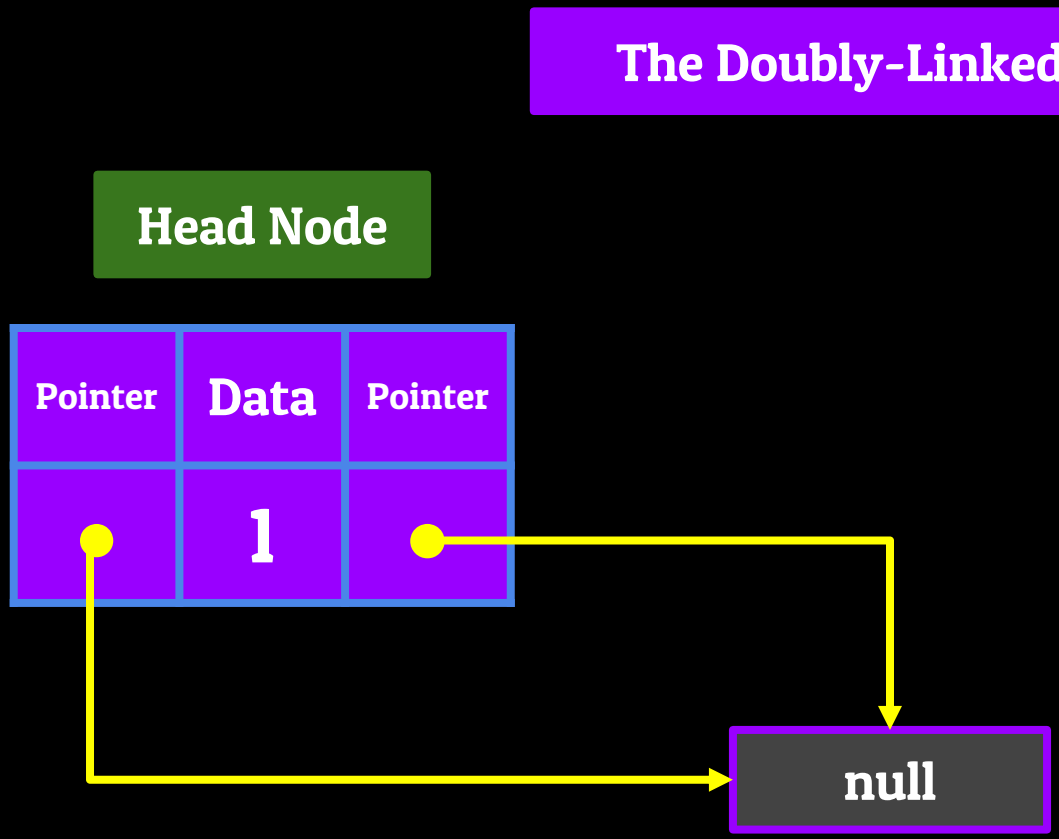
Head Node

Pointer	Data	Pointer
•	1	•

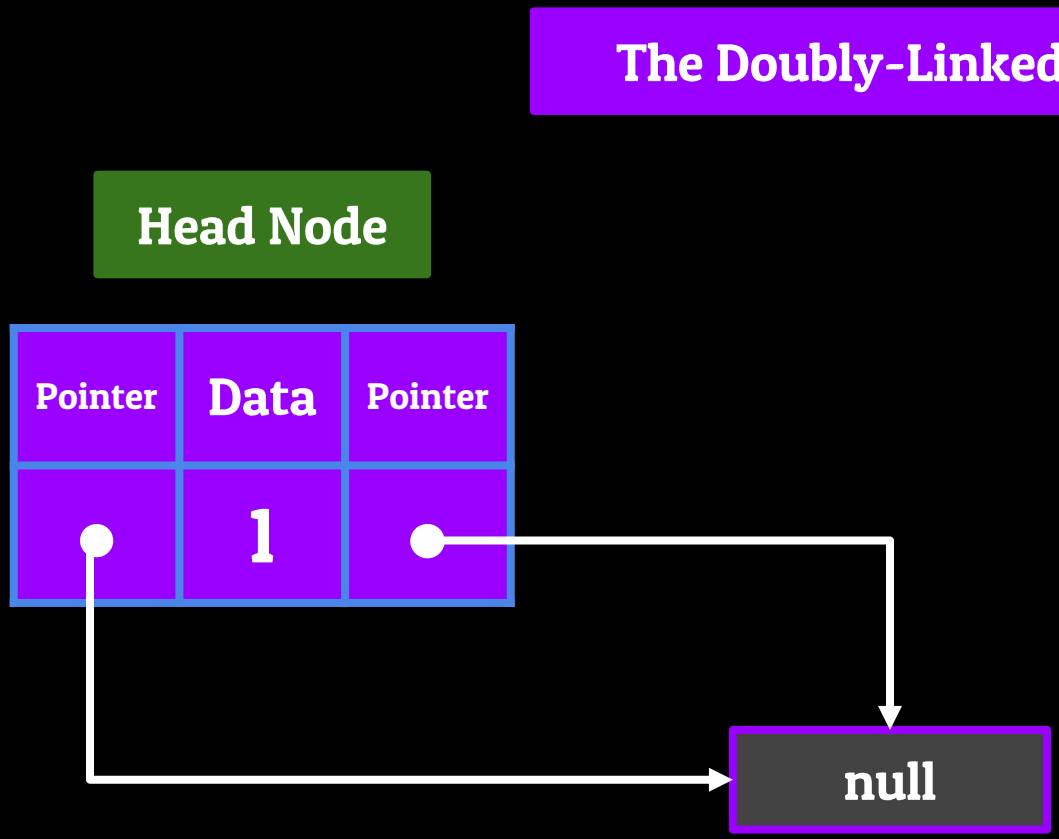
The Doubly-Linked List - Visualization



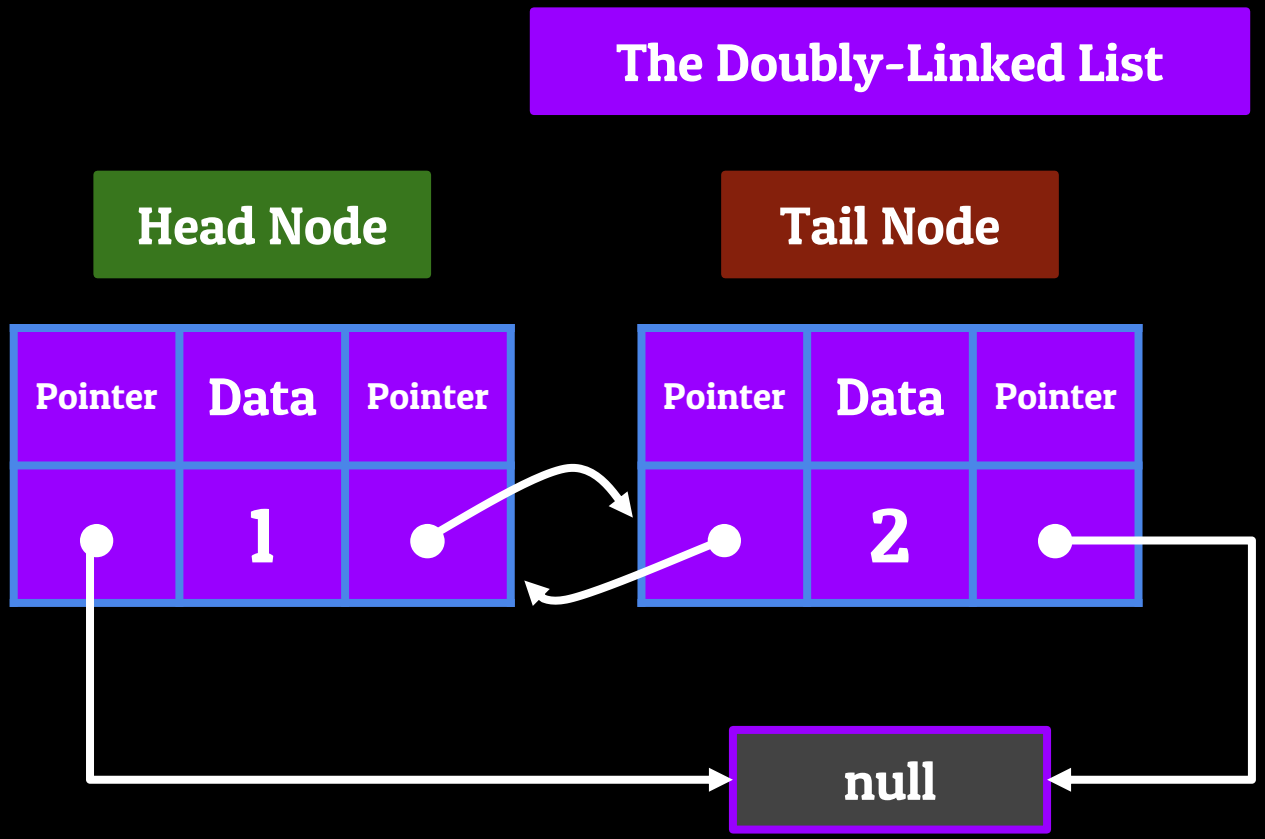
The Doubly-Linked List - Visualization



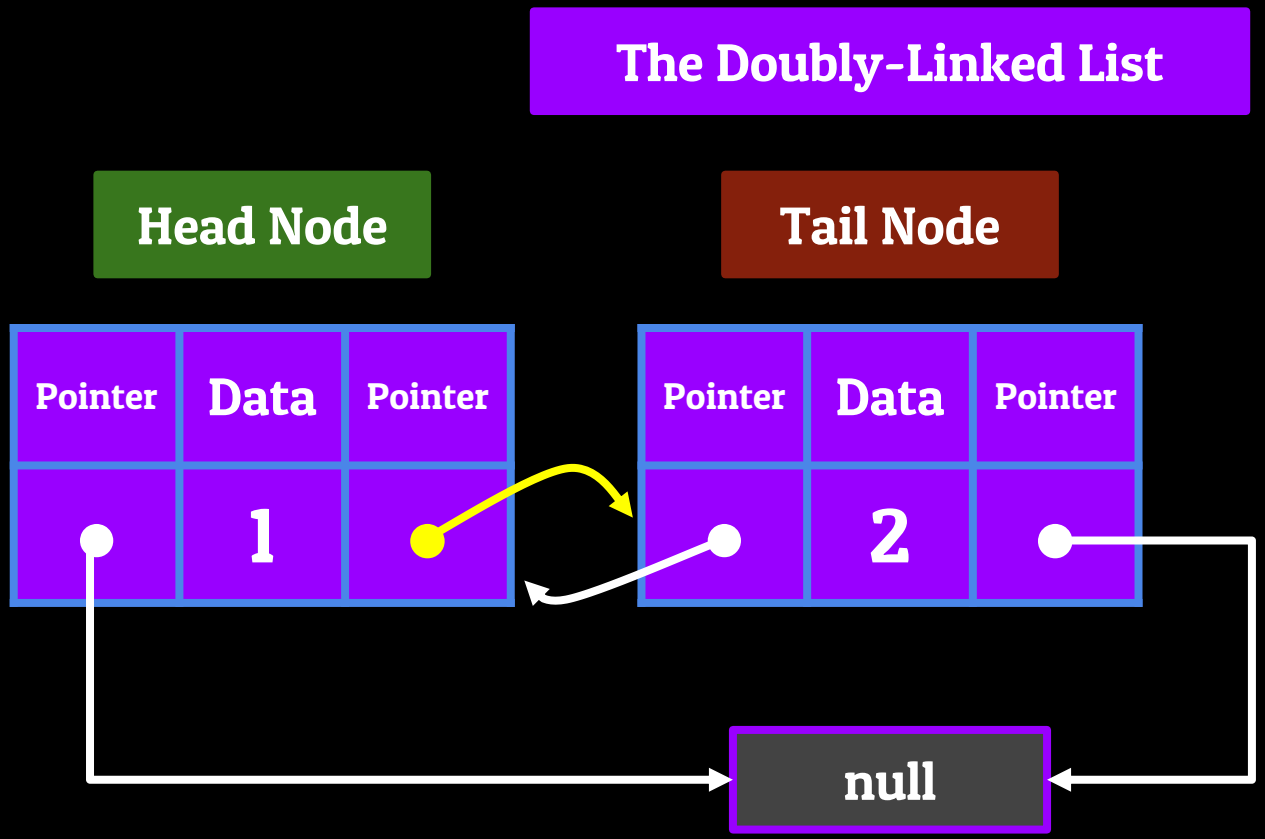
The Doubly-Linked List - Visualization



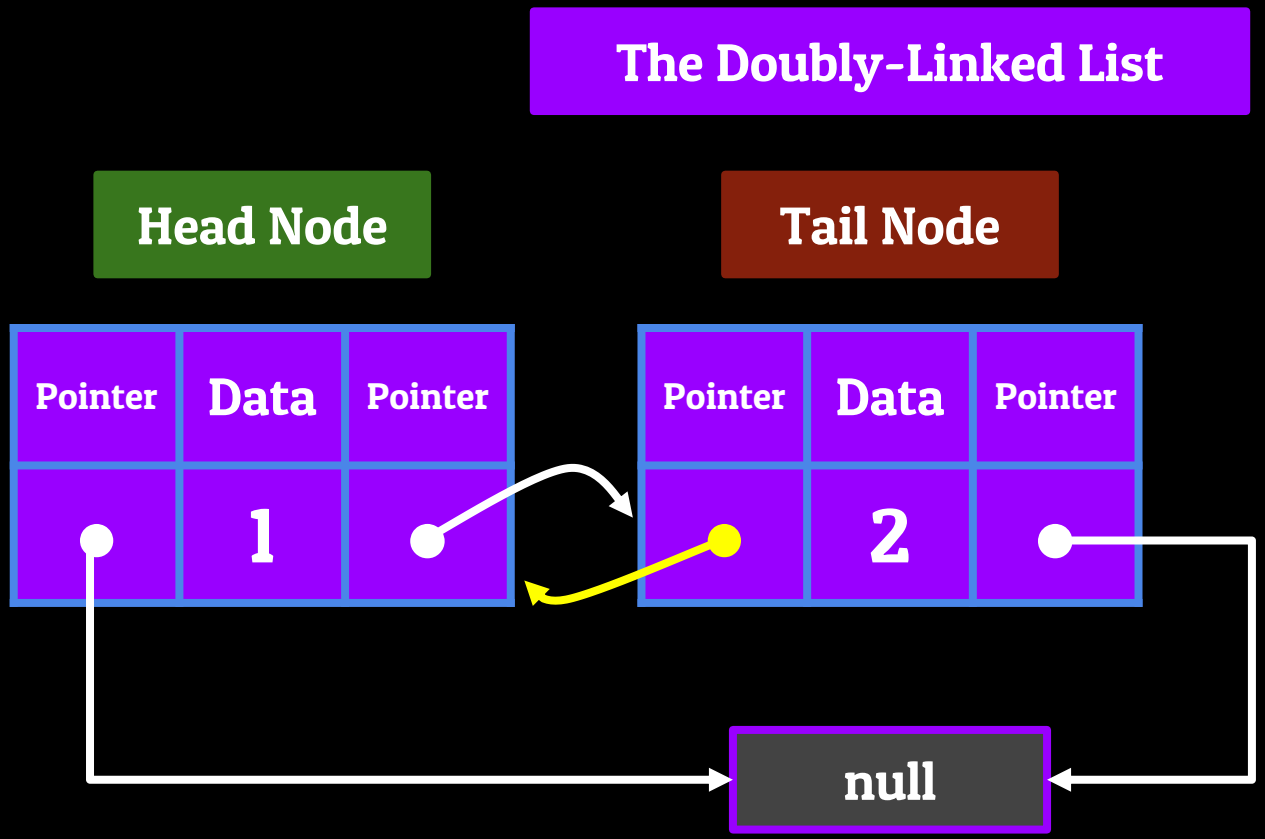
The Doubly-Linked List - Visualization



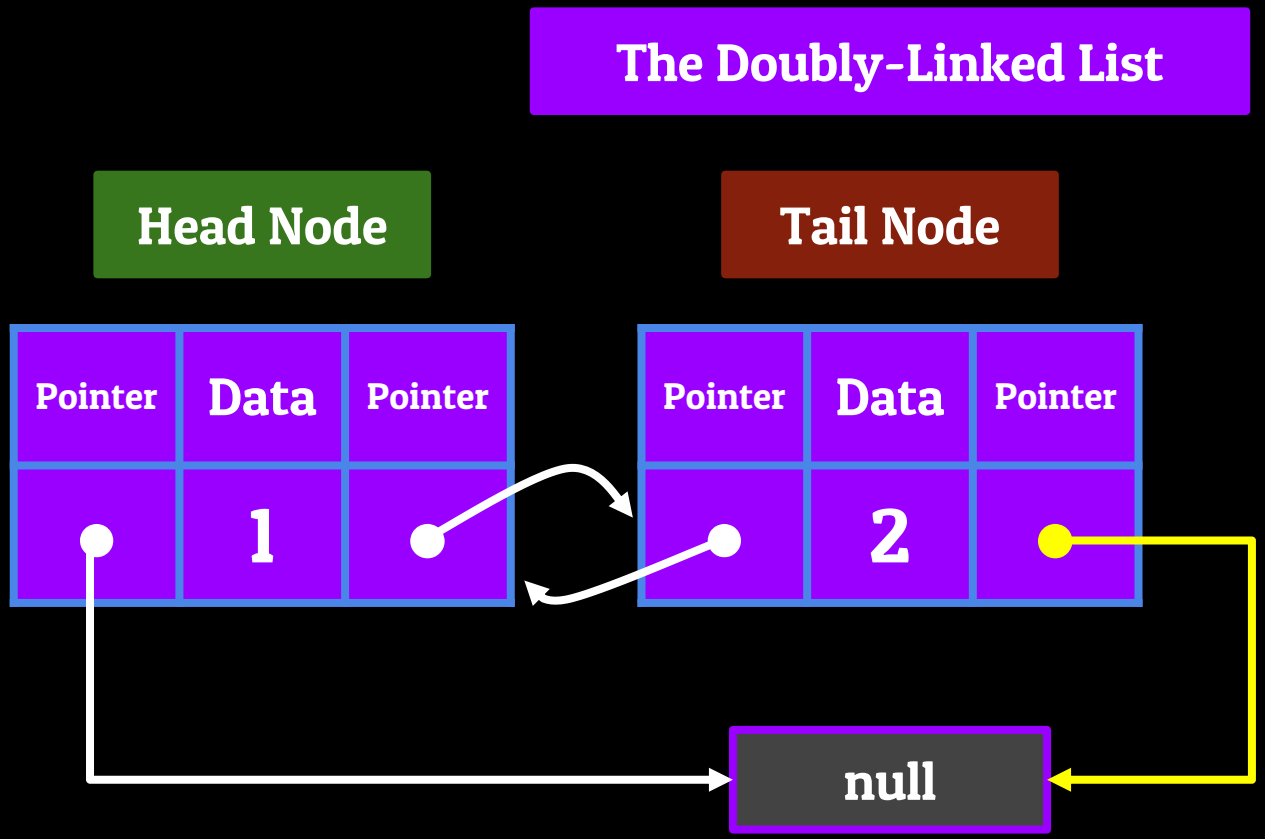
The Doubly-Linked List - Visualization



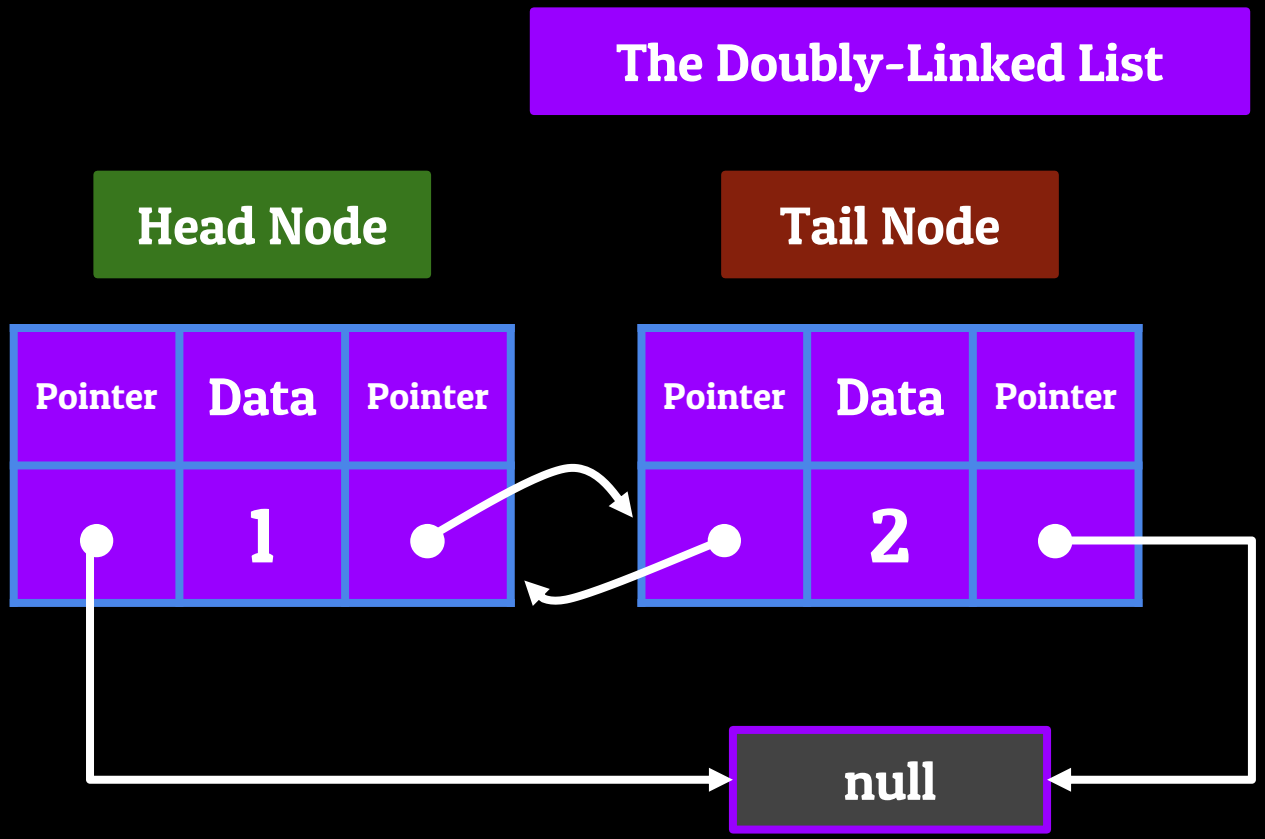
The Doubly-Linked List - Visualization



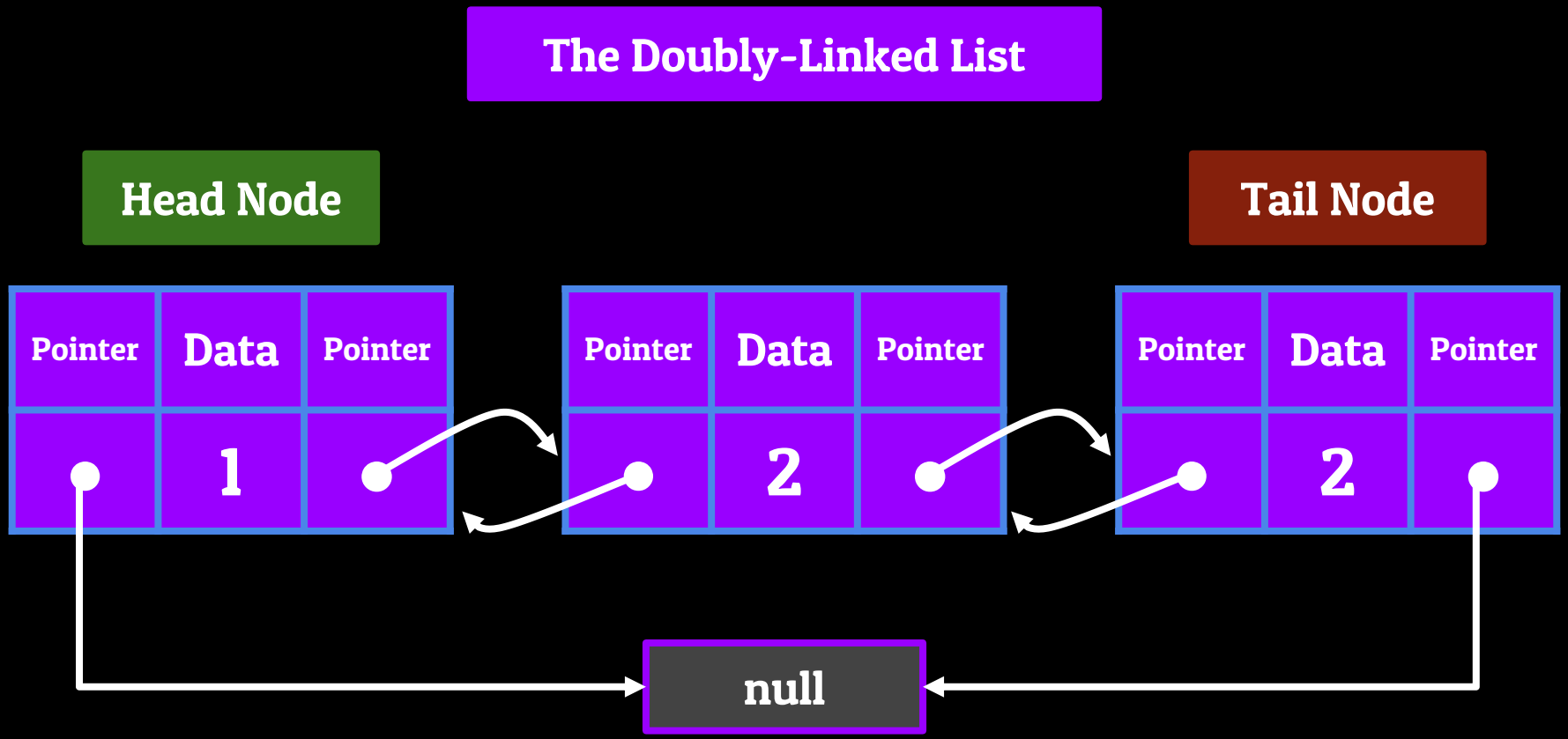
The Doubly-Linked List - Visualization



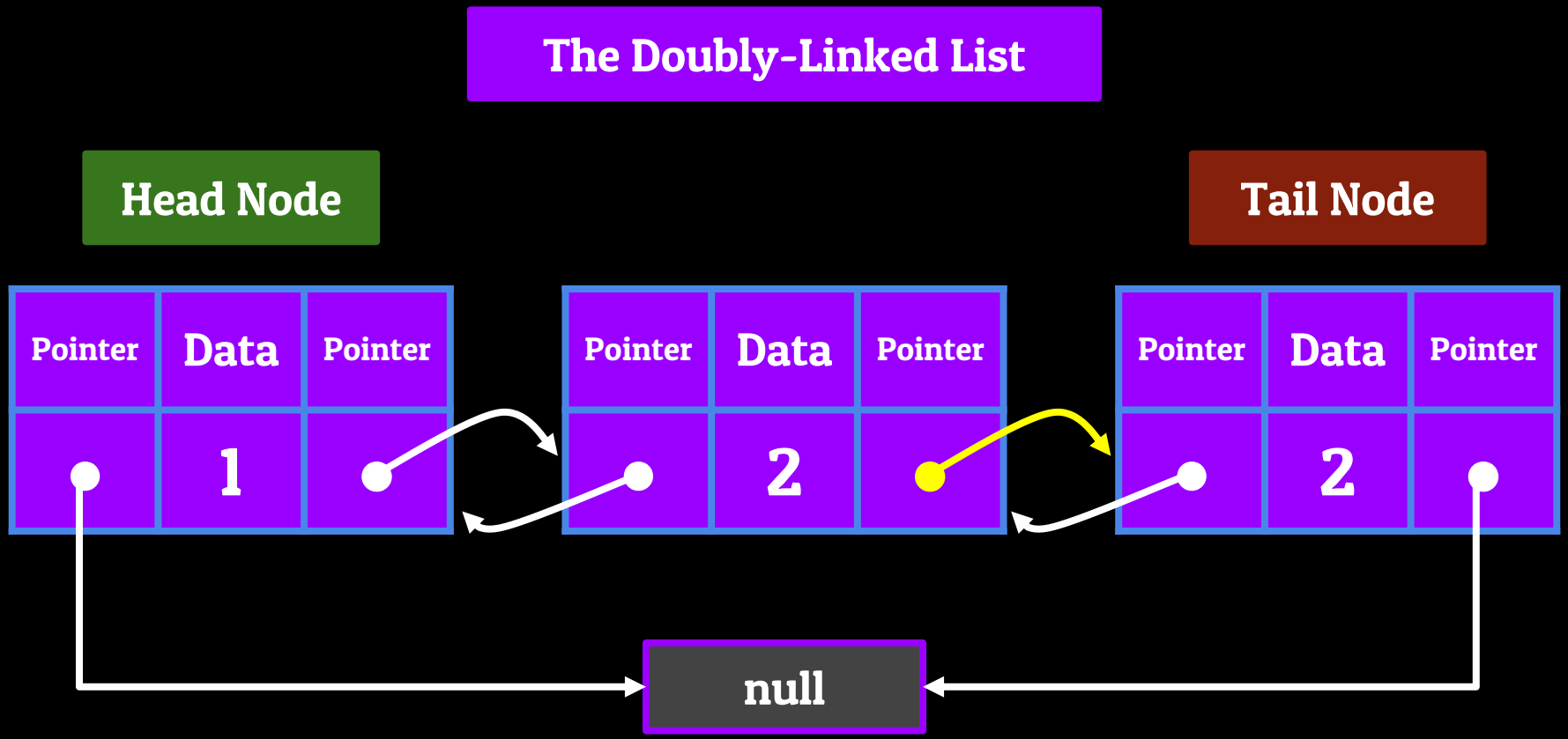
The Doubly-Linked List - Visualization



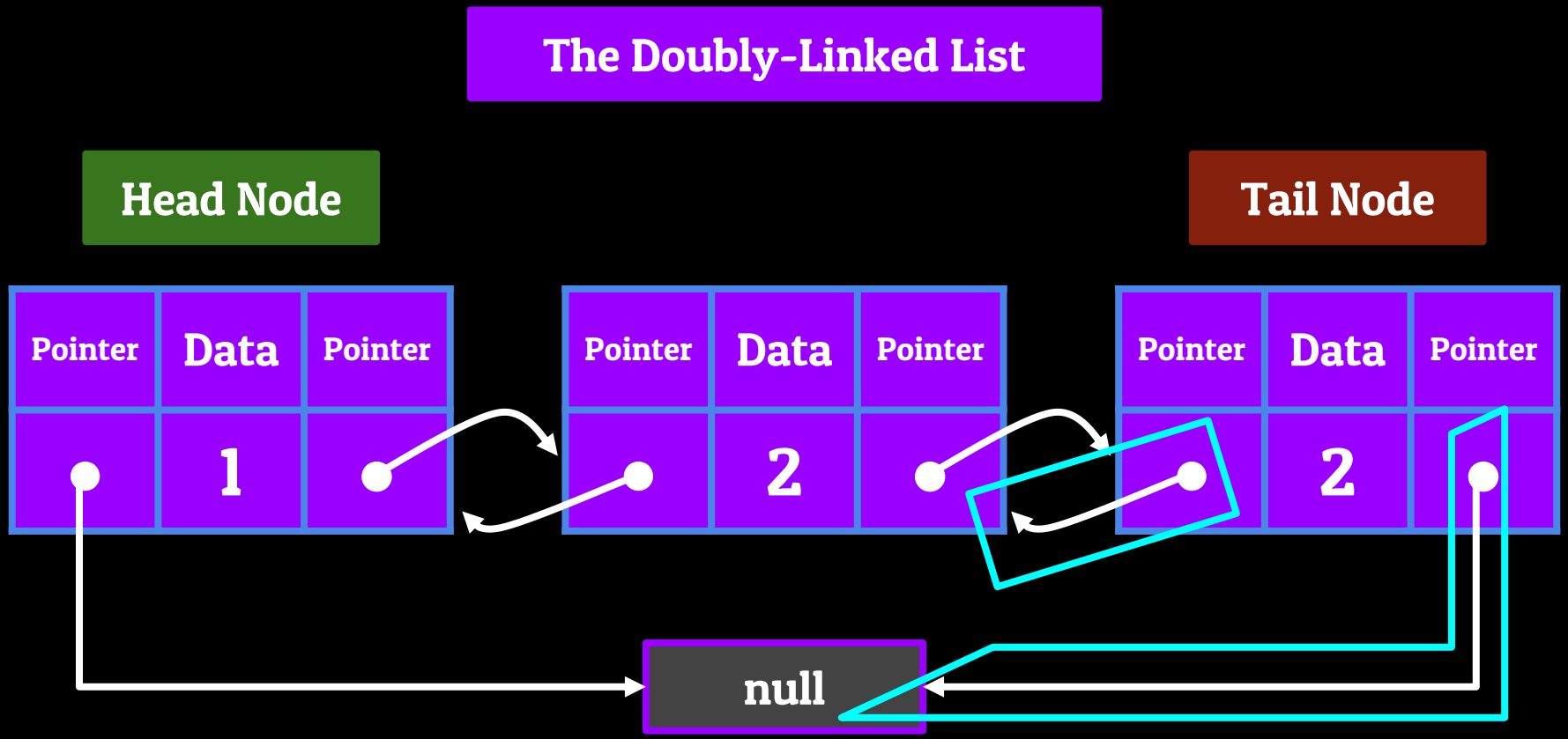
The Doubly-Linked List - Visualization



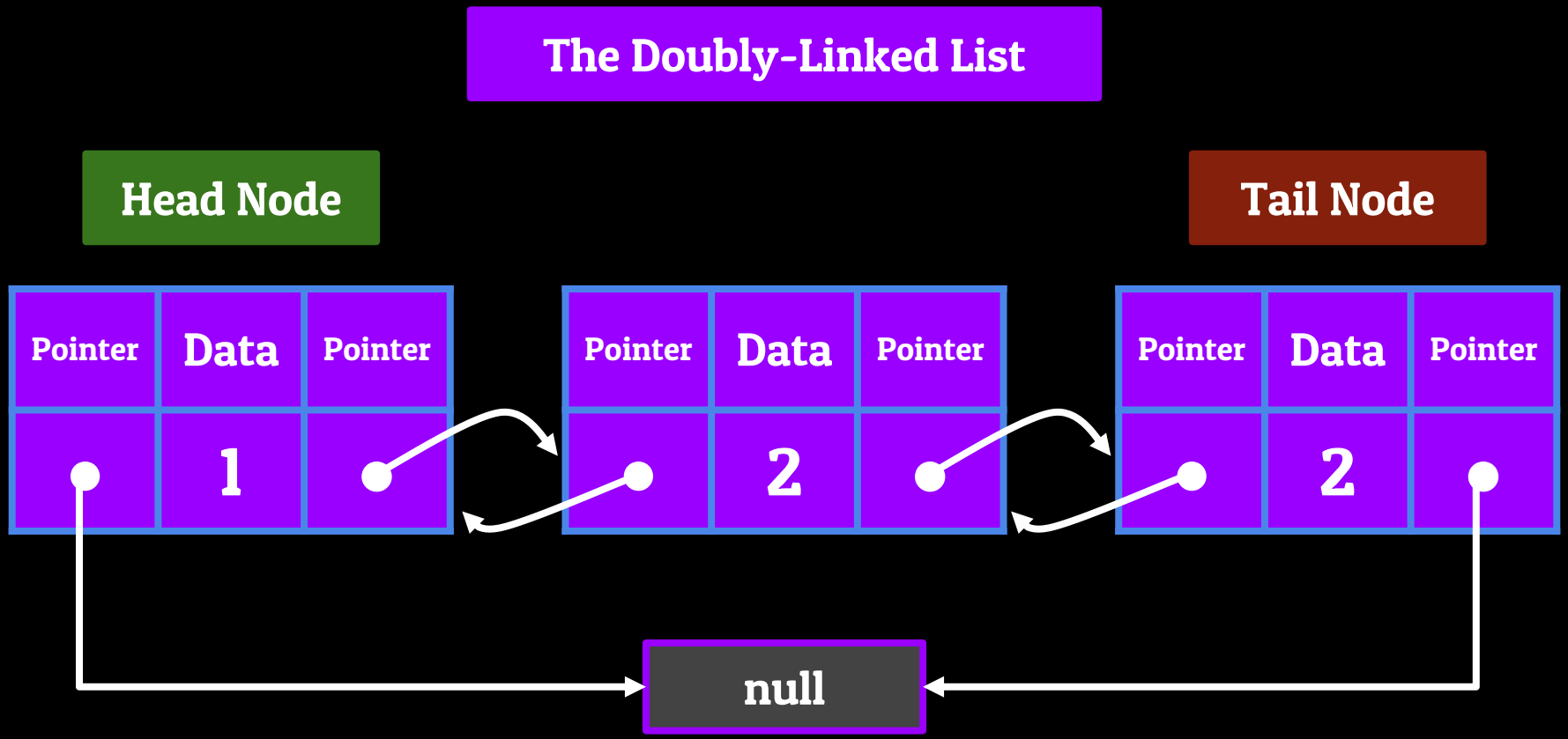
The Doubly-Linked List - Visualization



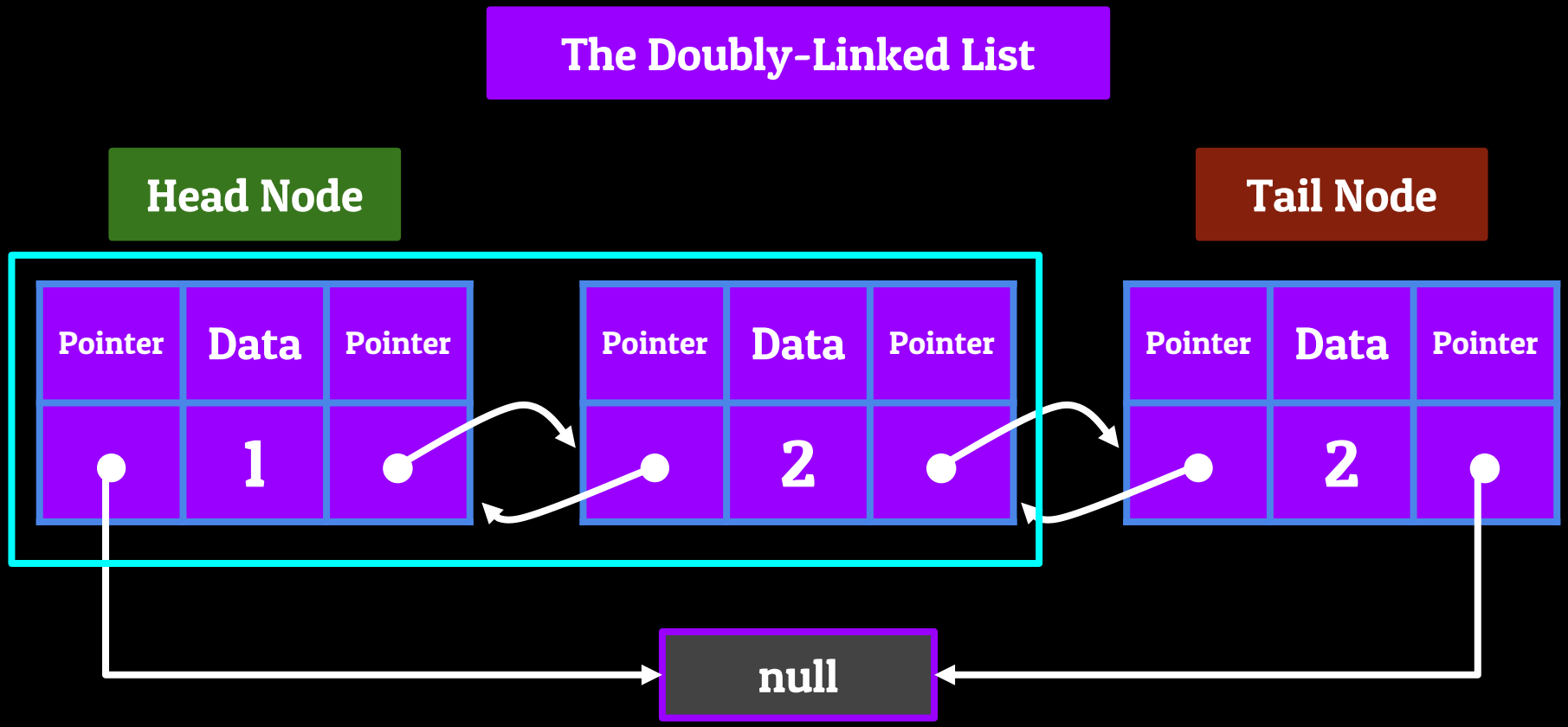
The Doubly-Linked List - Visualization



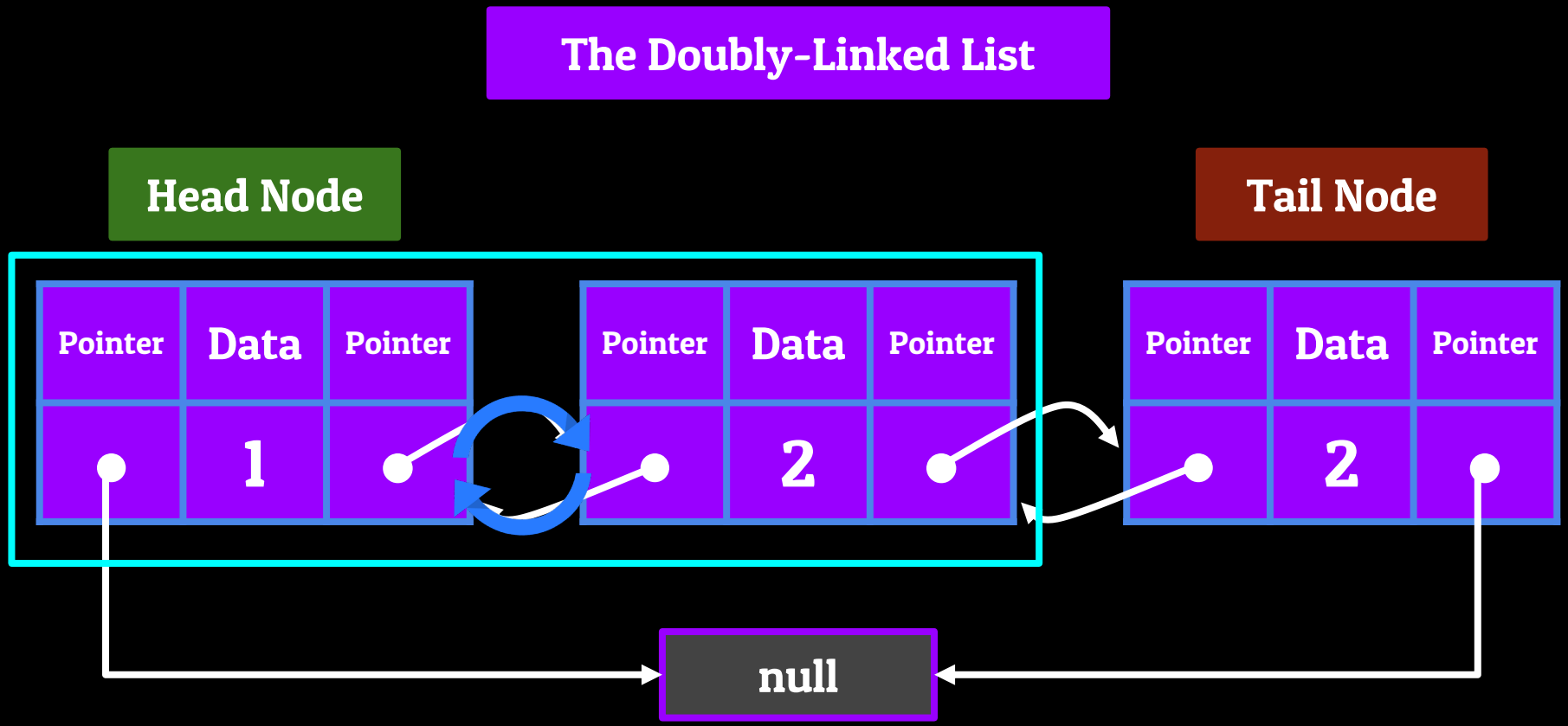
The Doubly-Linked List - Visualization



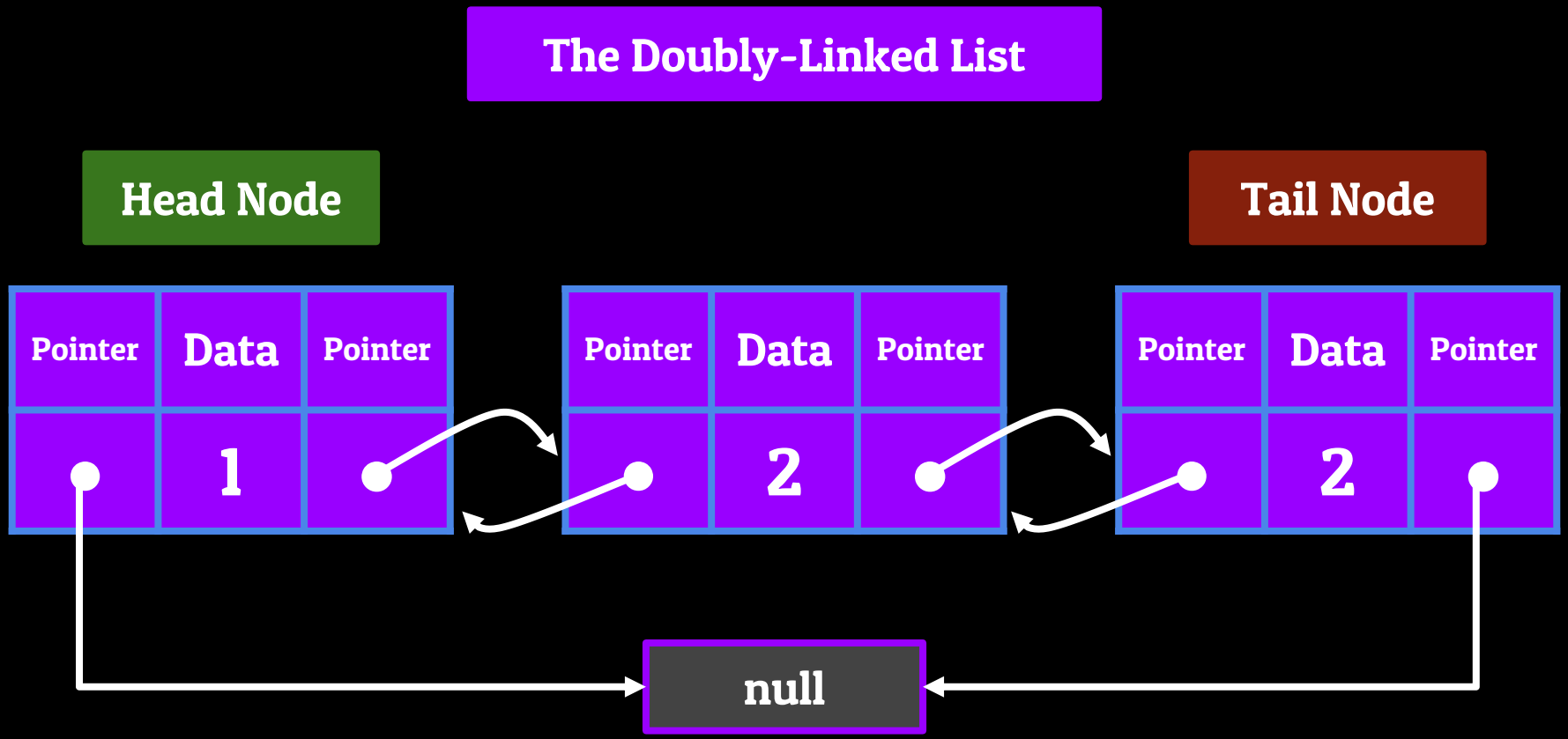
The Doubly-Linked List - Visualization



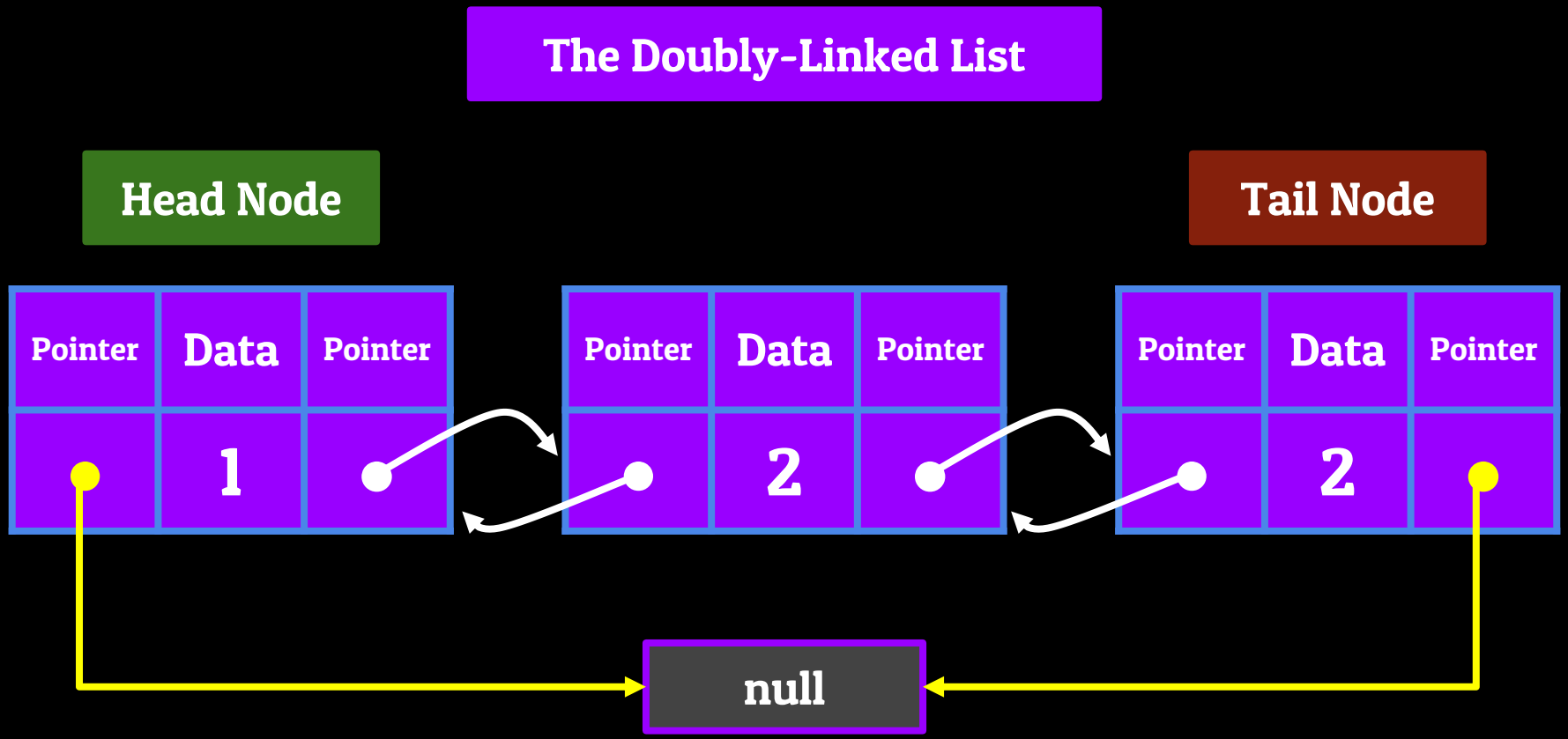
The Doubly-Linked List - Visualization



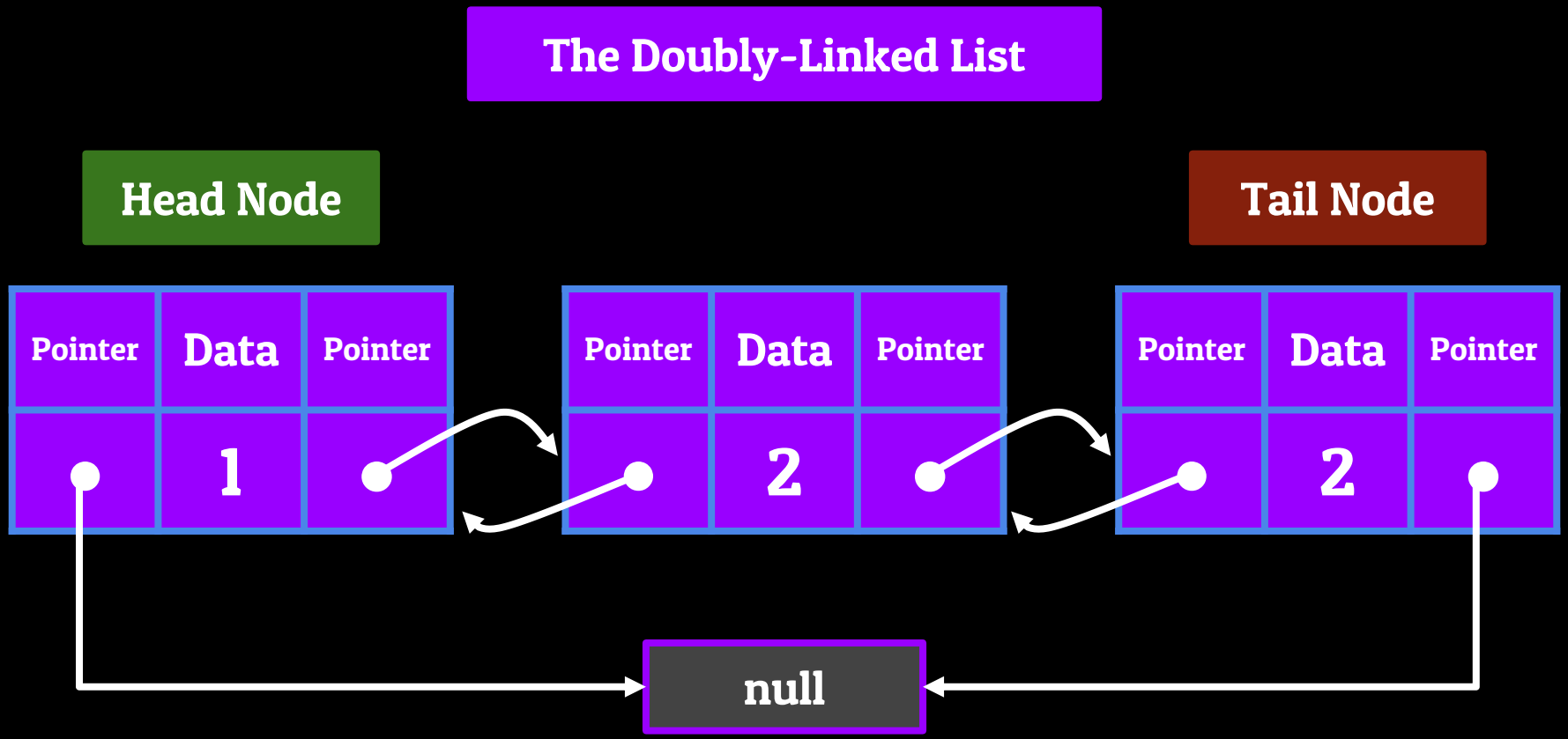
The Doubly-Linked List - Visualization



The Doubly-Linked List - Visualization



The Doubly-Linked List - Visualization



The Doubly-Linked List - Adding and Removing Information

The Doubly-Linked List - Adding and Removing Information

Add to Head

Remove from Head

The Doubly-Linked List - Adding and Removing Information

Add to Head

Add to the Middle

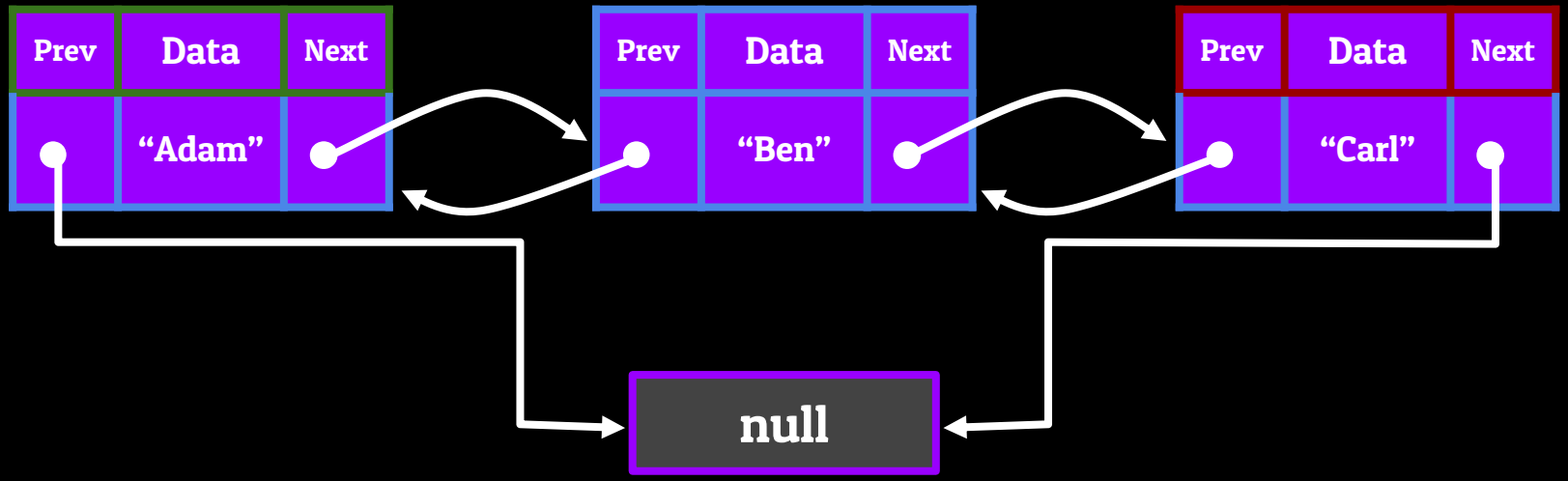
Remove from Head

Remove from the Middle

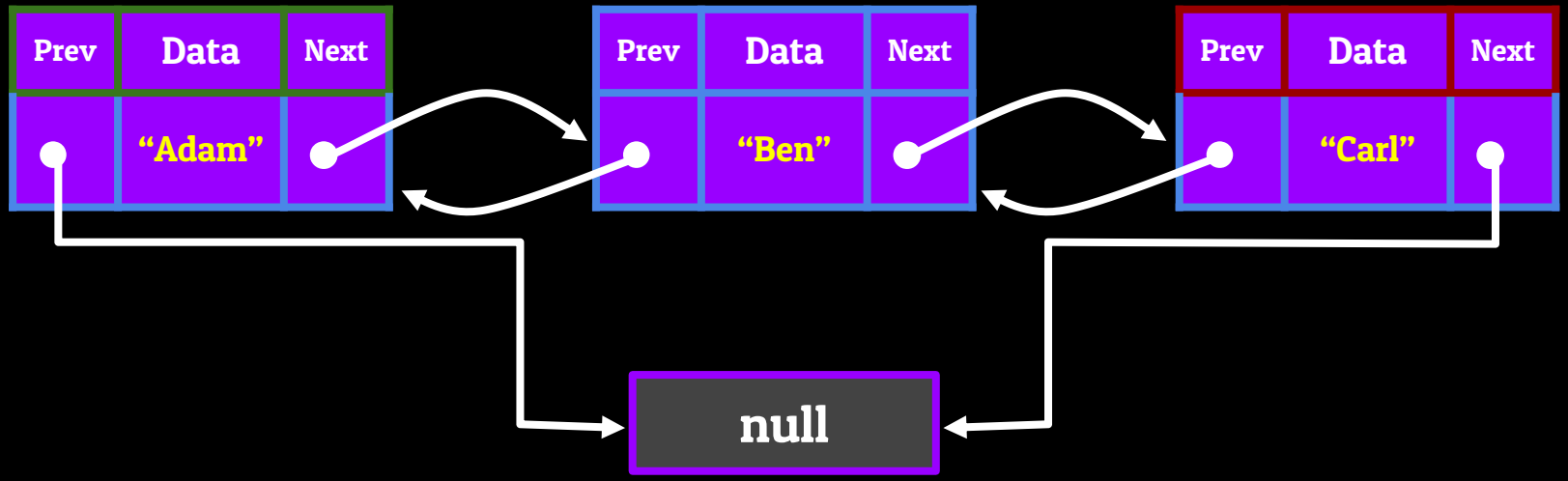
The Doubly-Linked List - Adding and Removing Information

Add to Head	Add to the Middle	Add to Tail
Remove from Head	Remove from the Middle	Remove from Tail

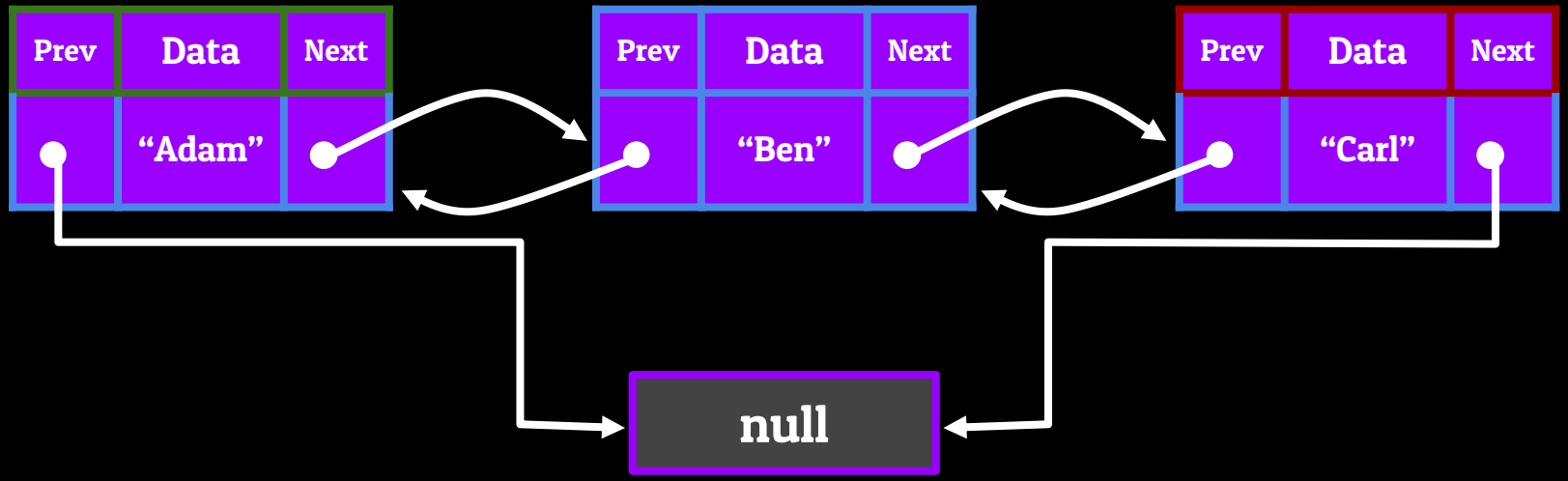
The Doubly-Linked List - Adding and Removing Information



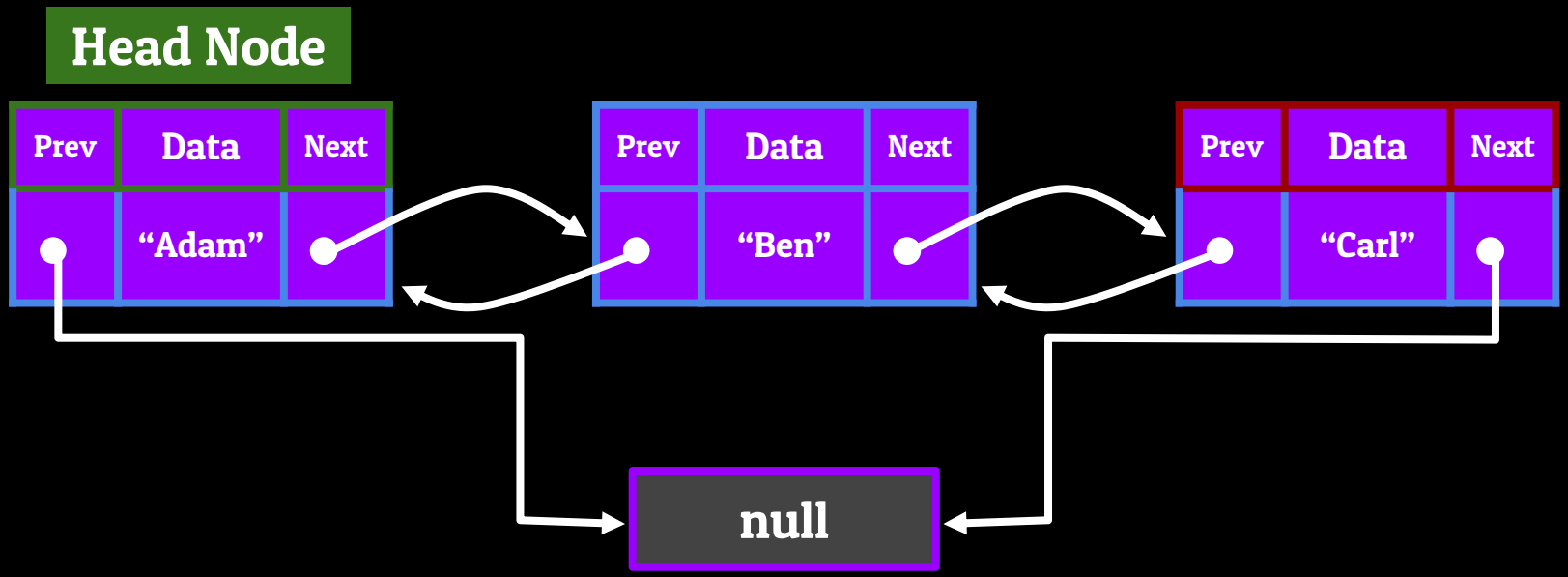
The Doubly-Linked List - Adding and Removing Information



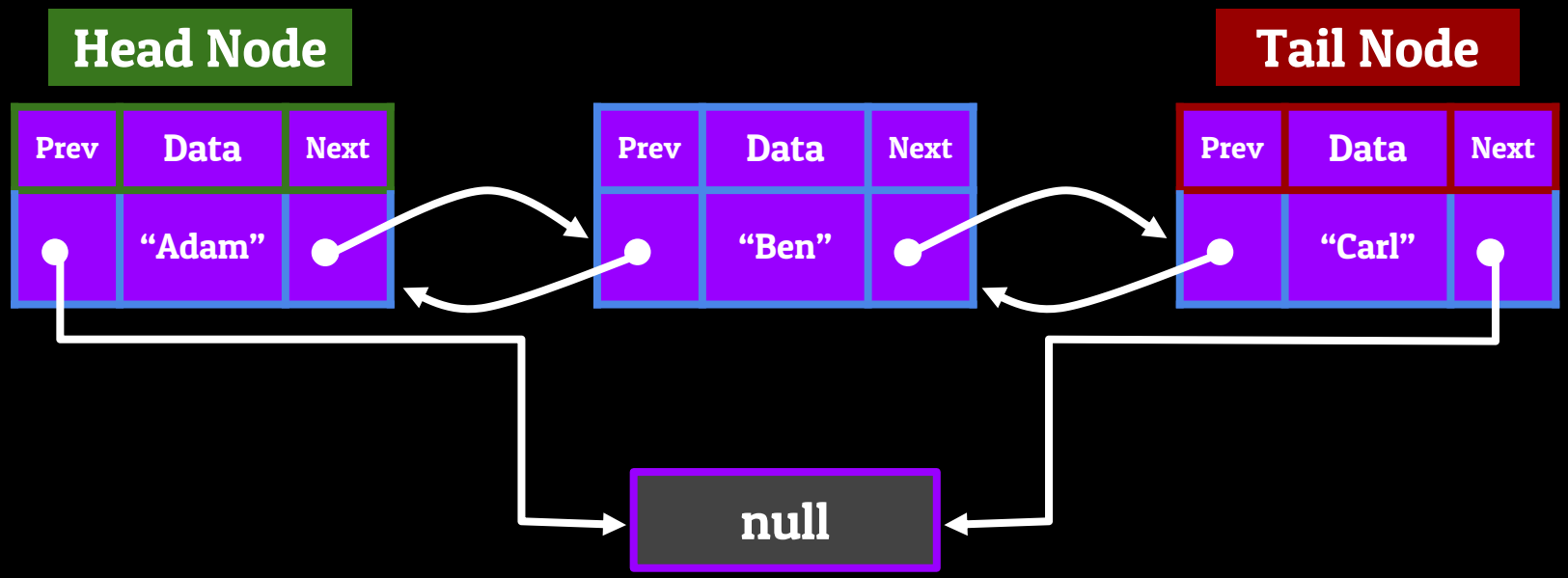
The Doubly-Linked List - Adding and Removing Information



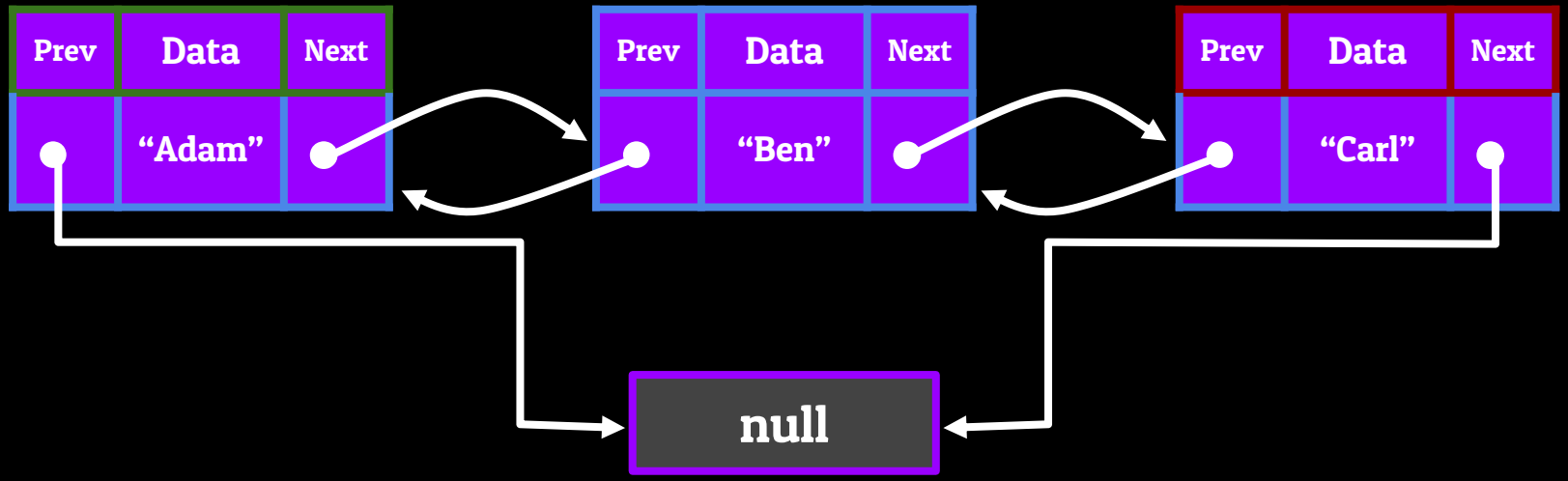
The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

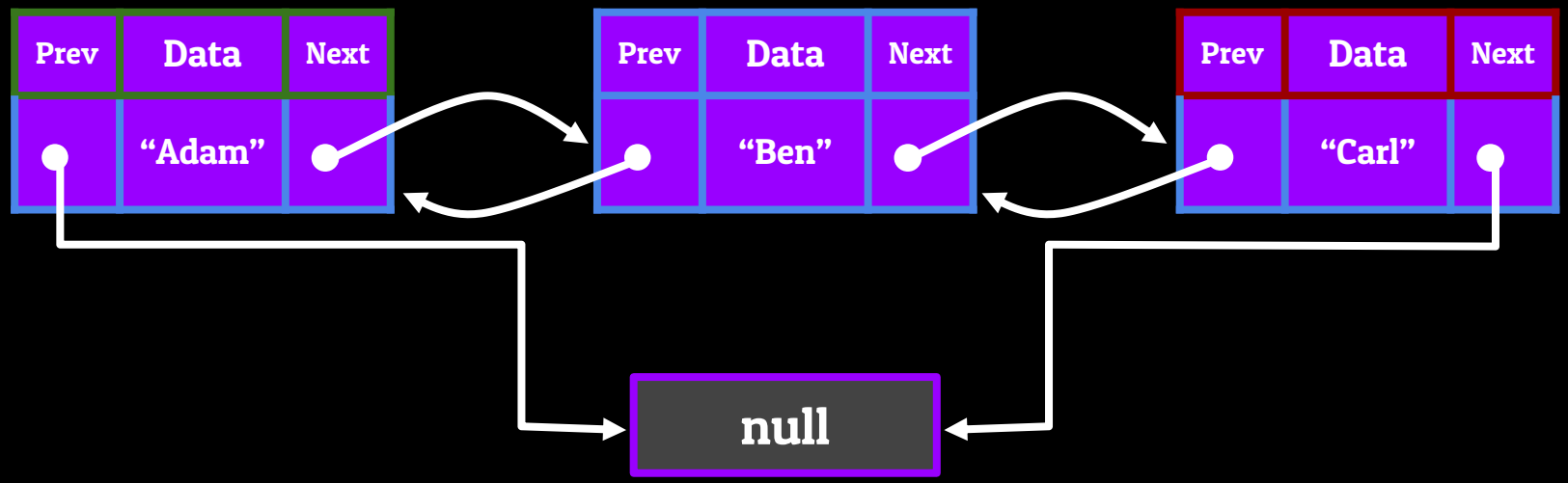


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

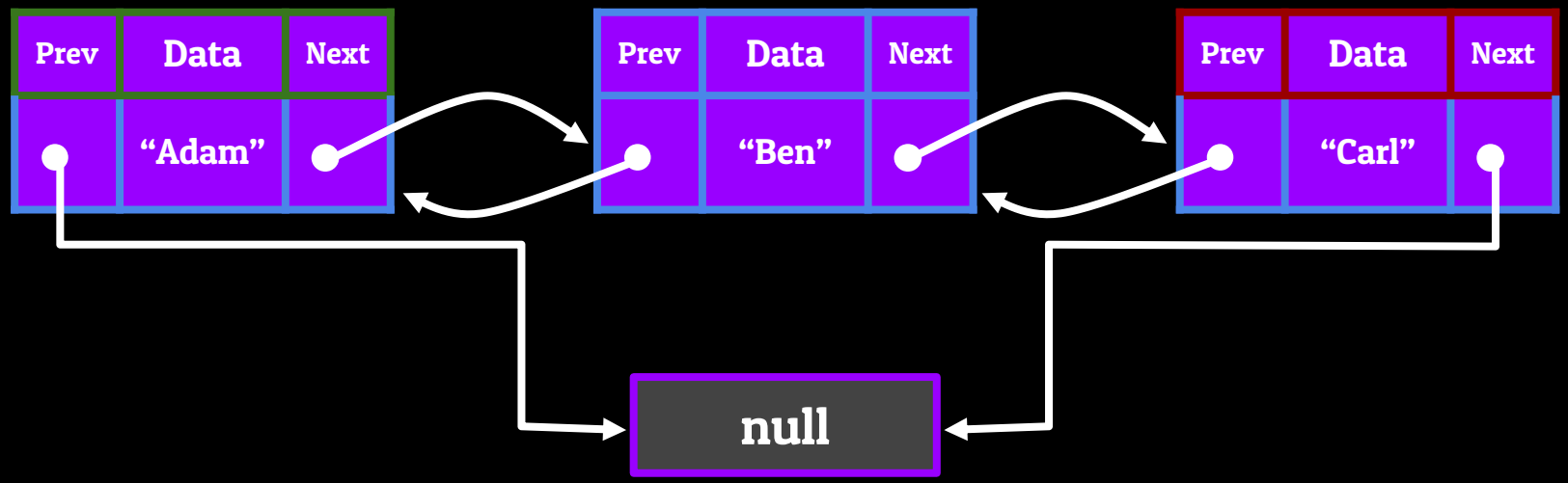
Adding to the Head of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

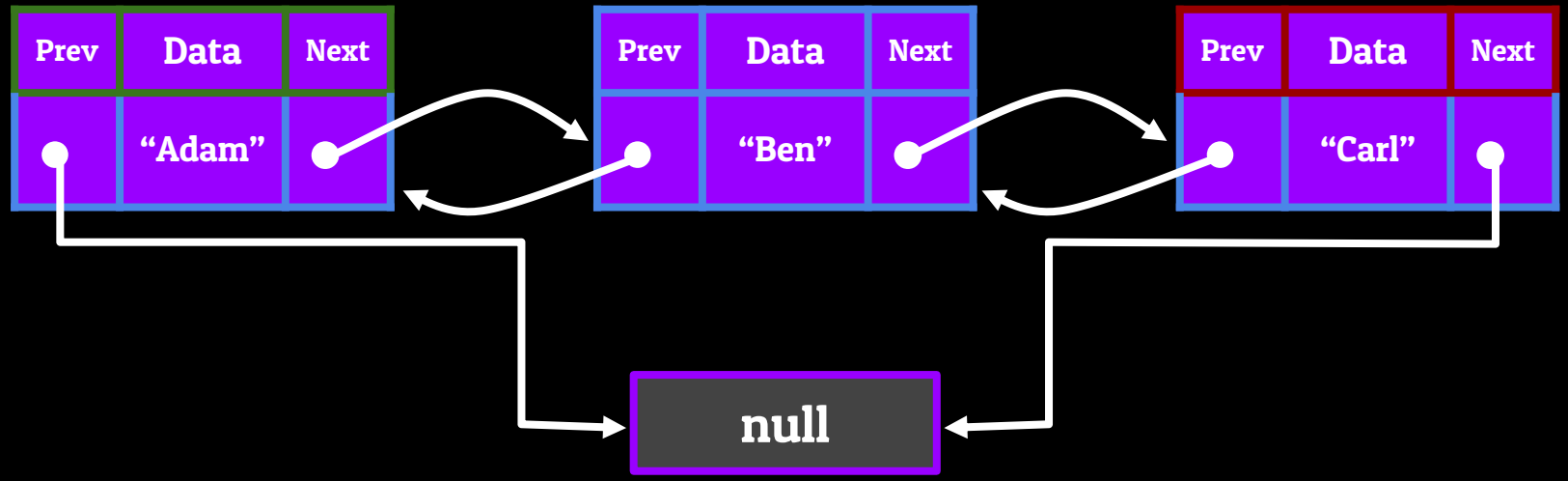


The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null



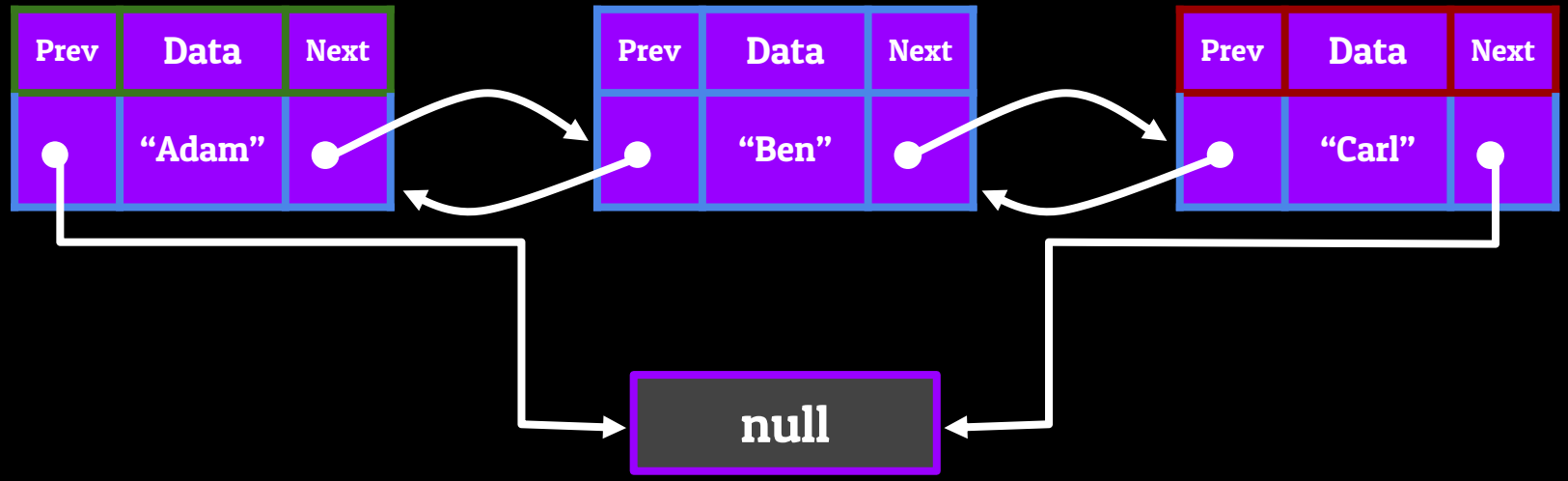
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



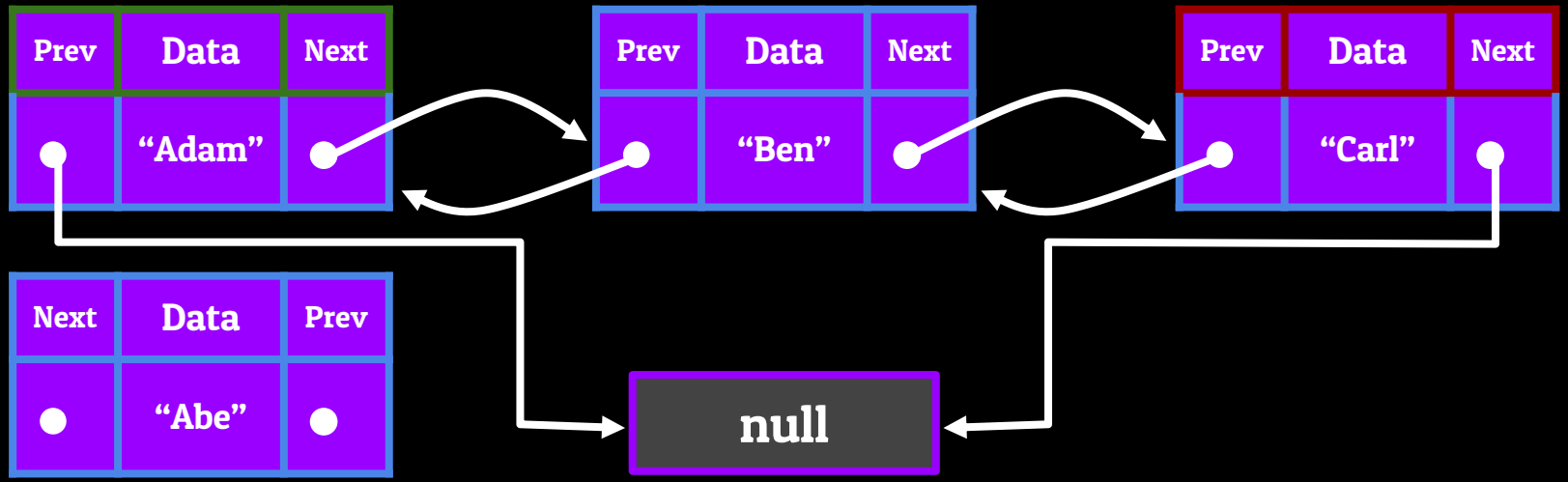
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



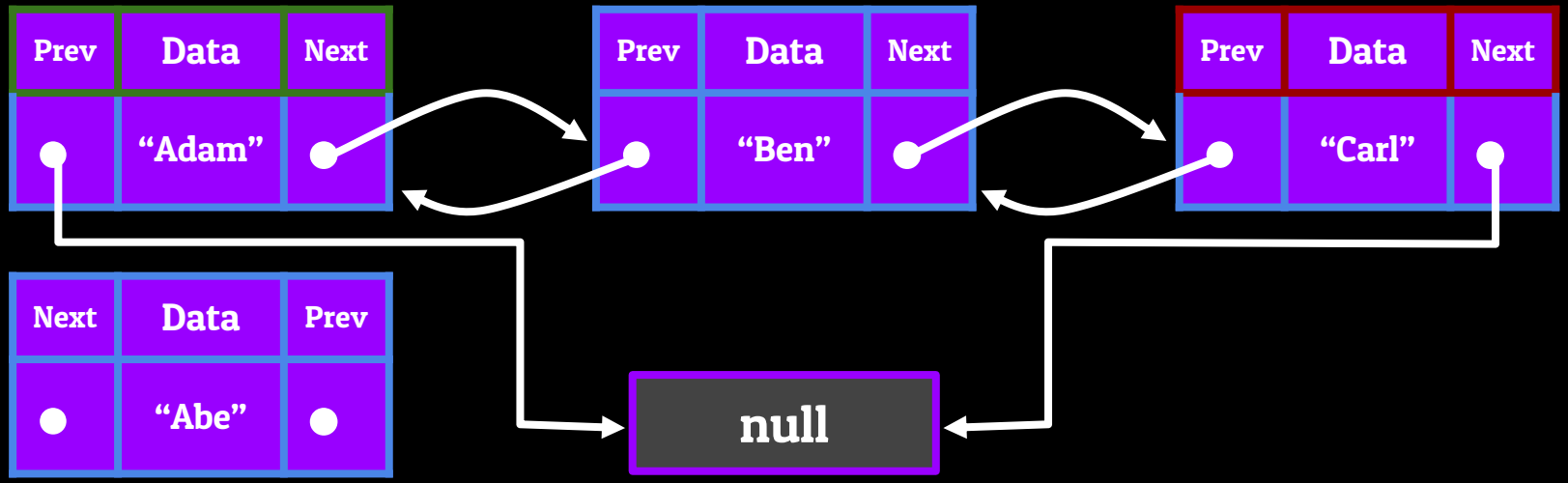
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



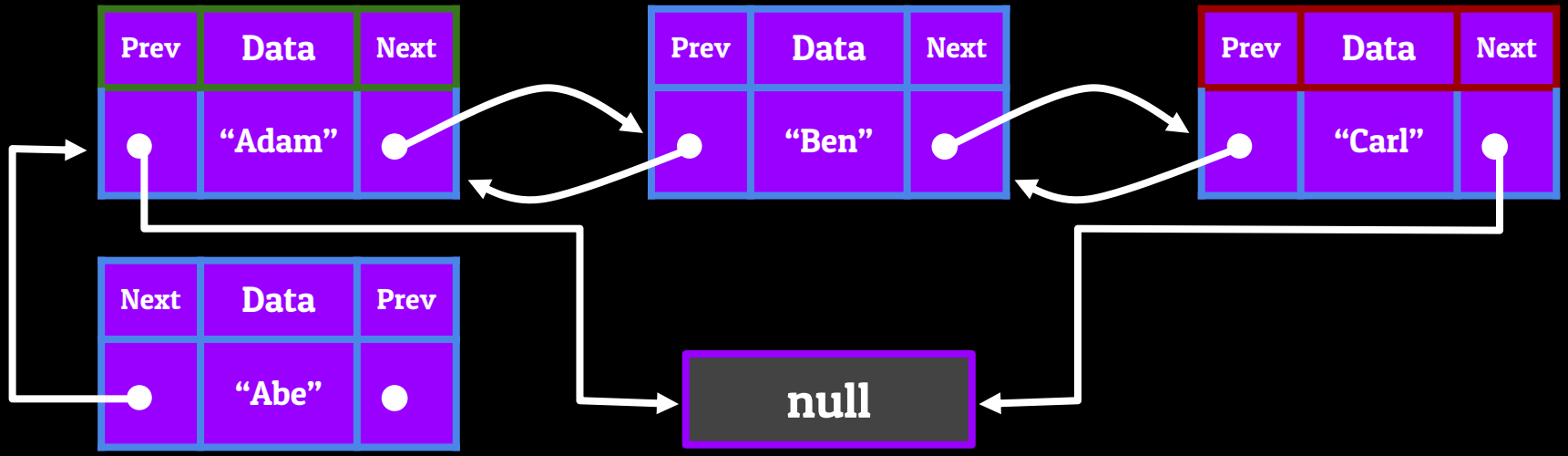
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



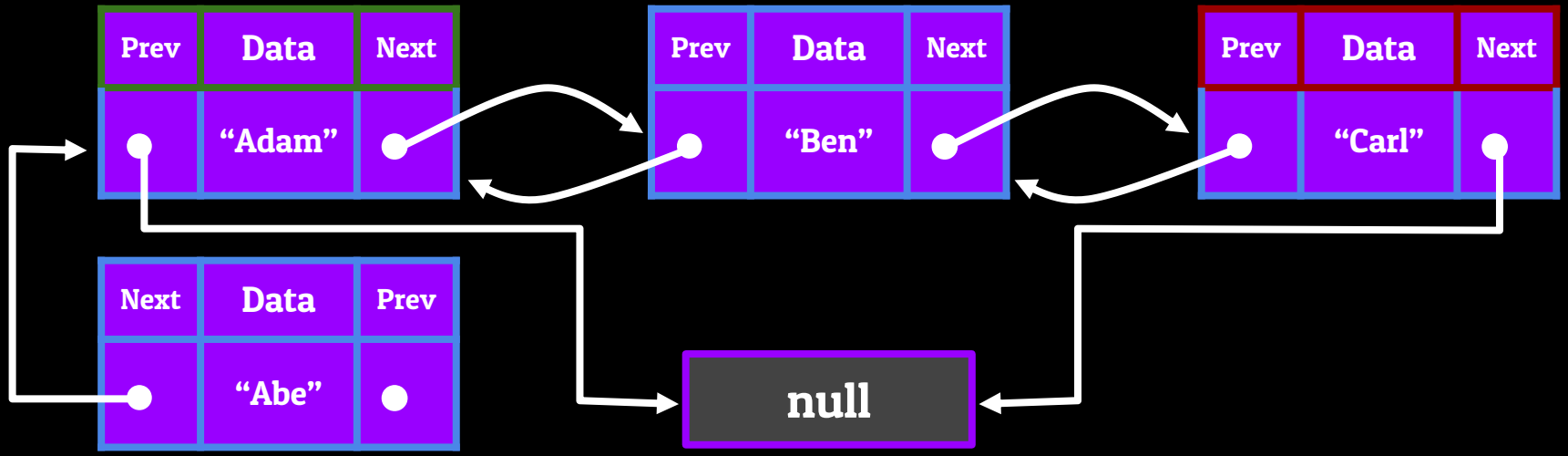
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



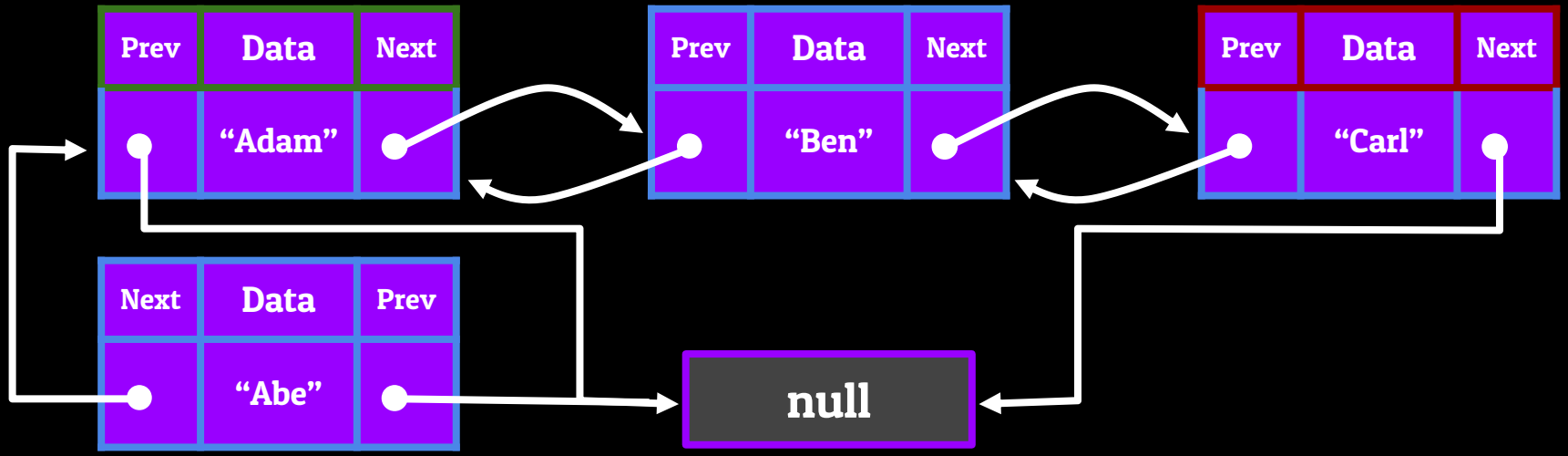
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



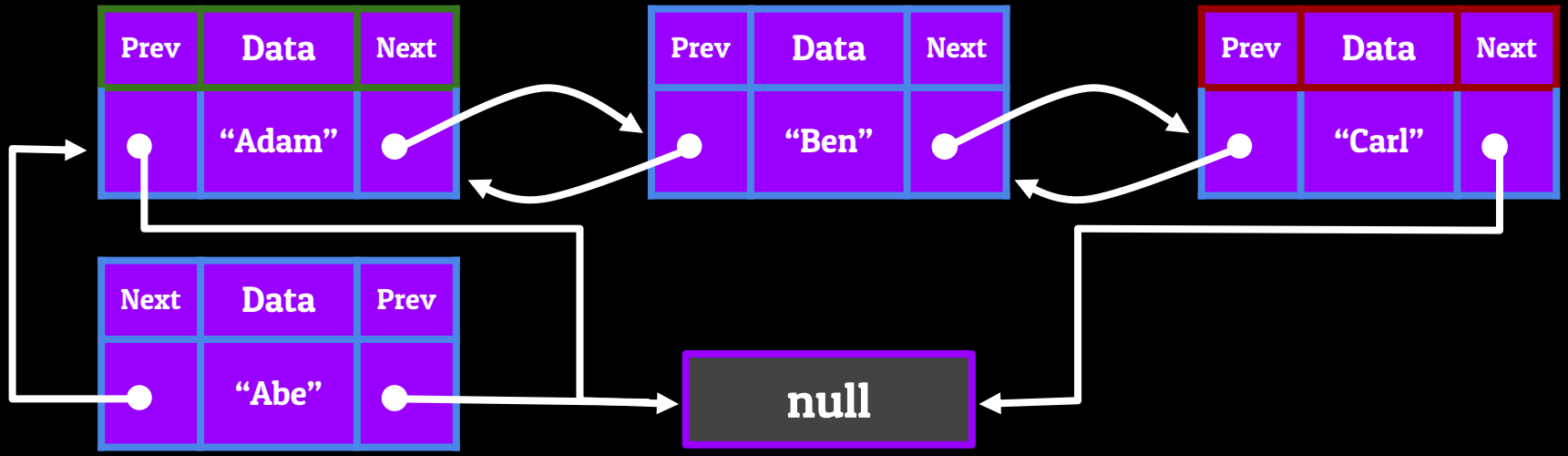
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



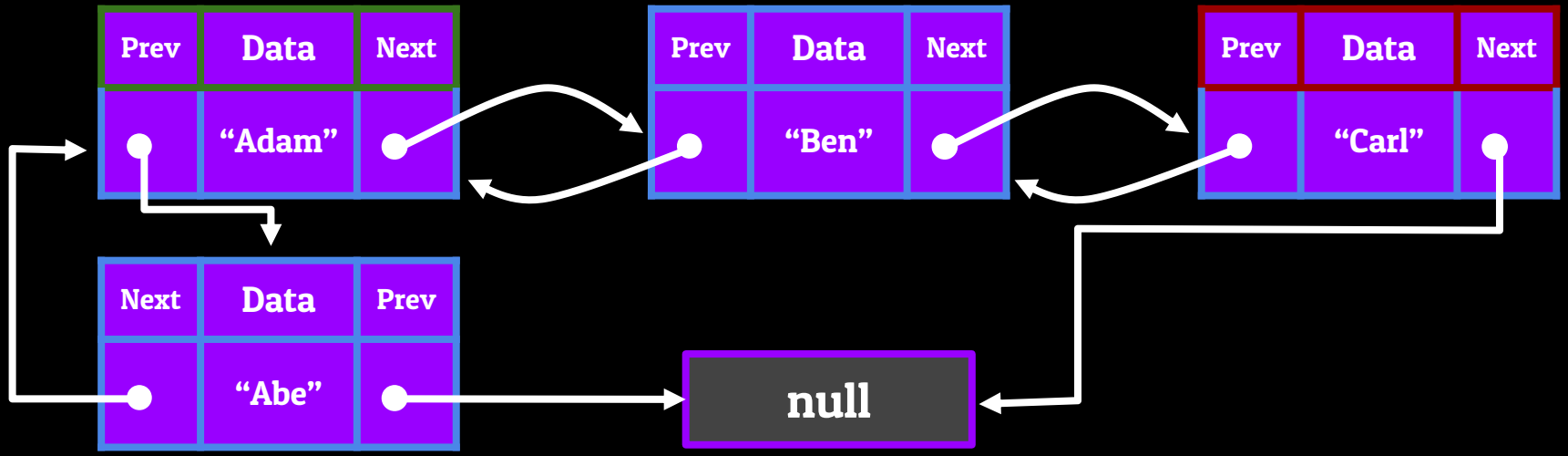
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node



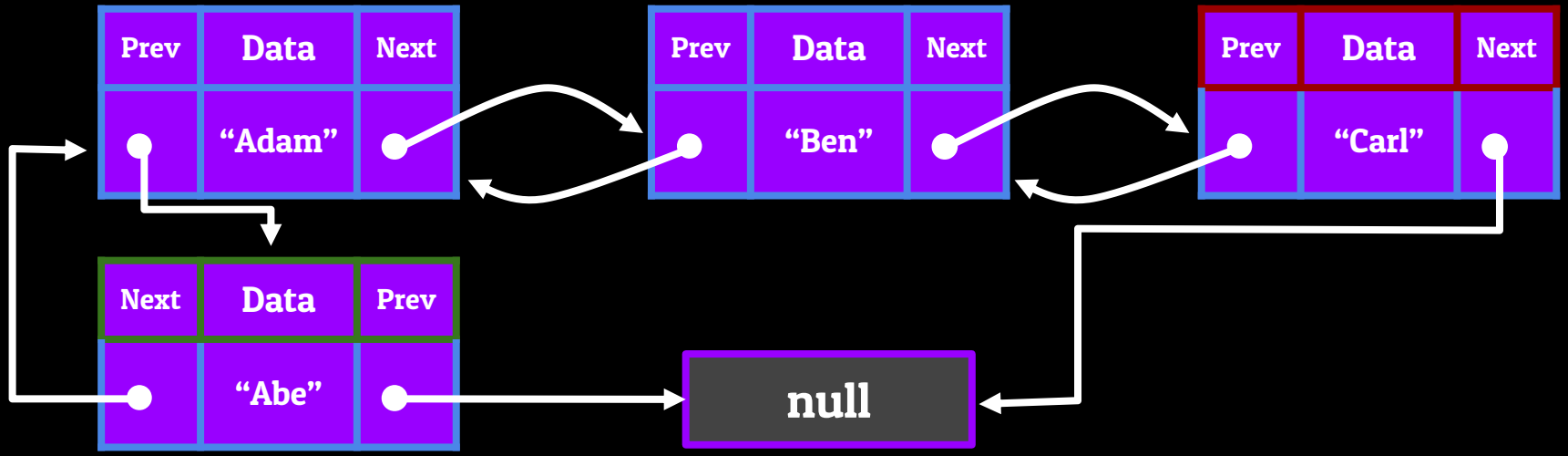
The Doubly-Linked List - Adding and Removing Information

Adding to the Head of a Doubly-LinkedList

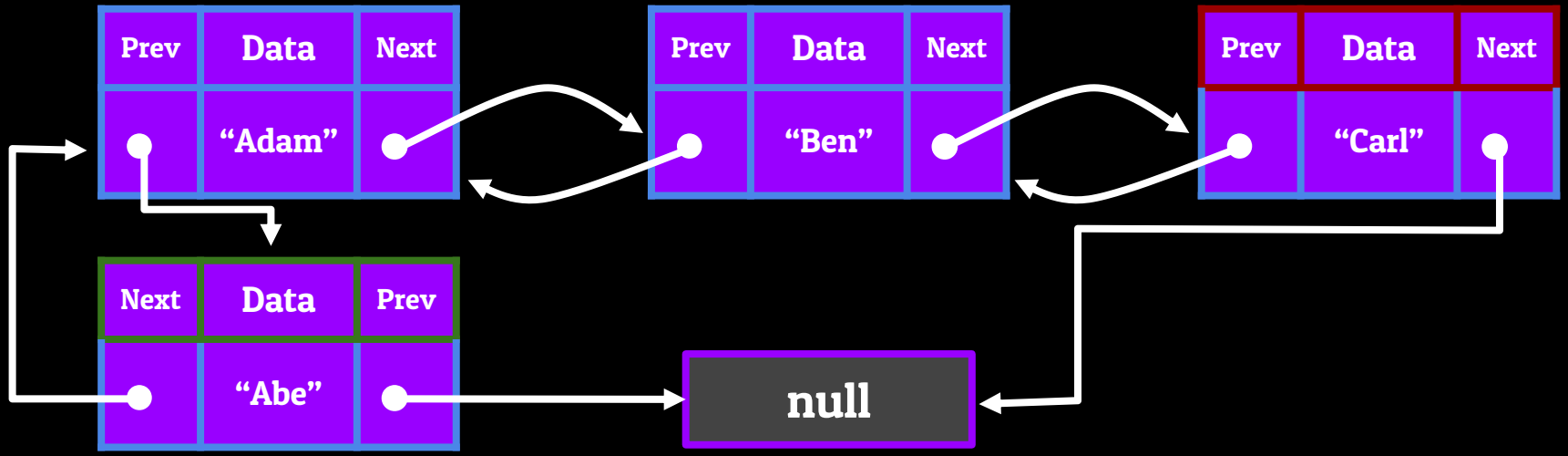
Set the new Nodes next to point towards the current head of the List

Take the new Node that we want to insert, and set it's previous to null

Set the current head's previous to point towards this new Node

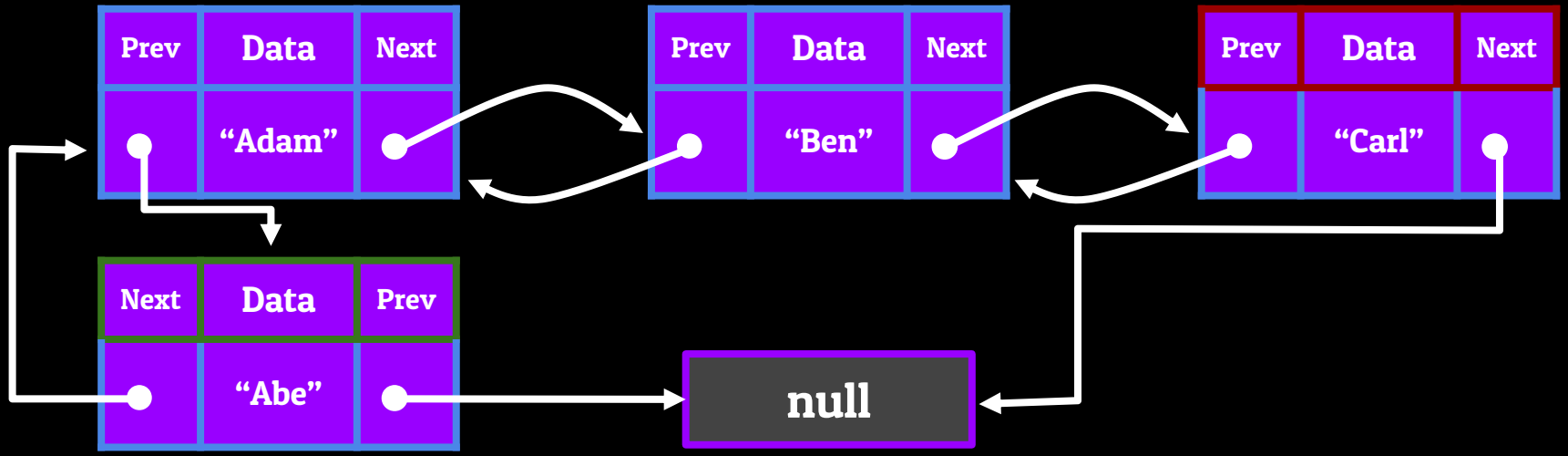


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

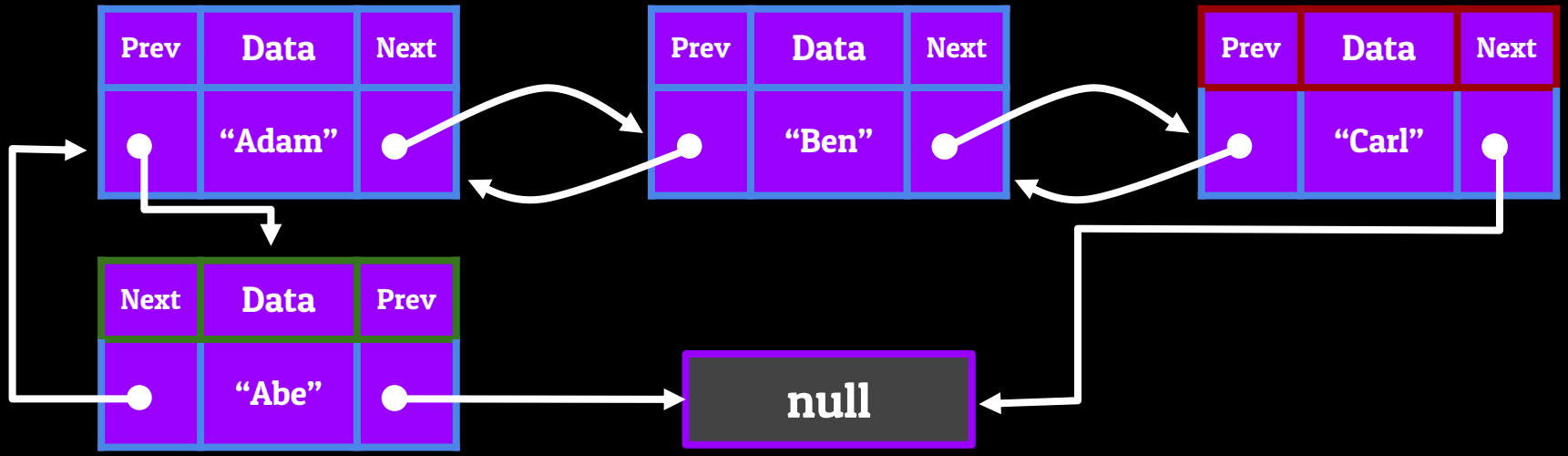
Removing from the Head of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

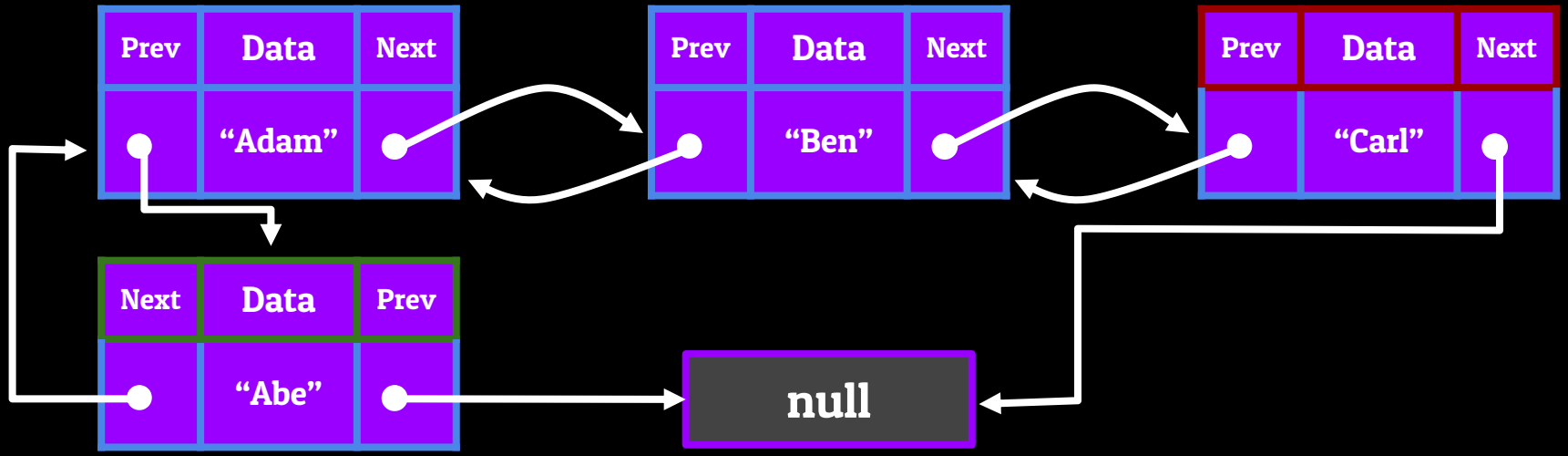


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

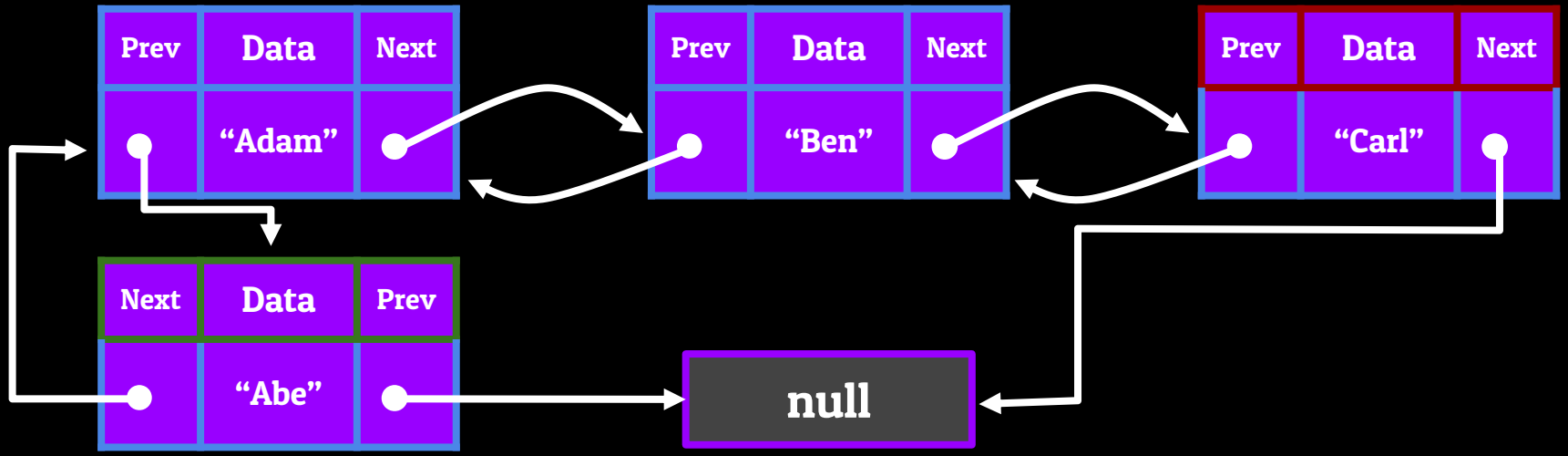


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

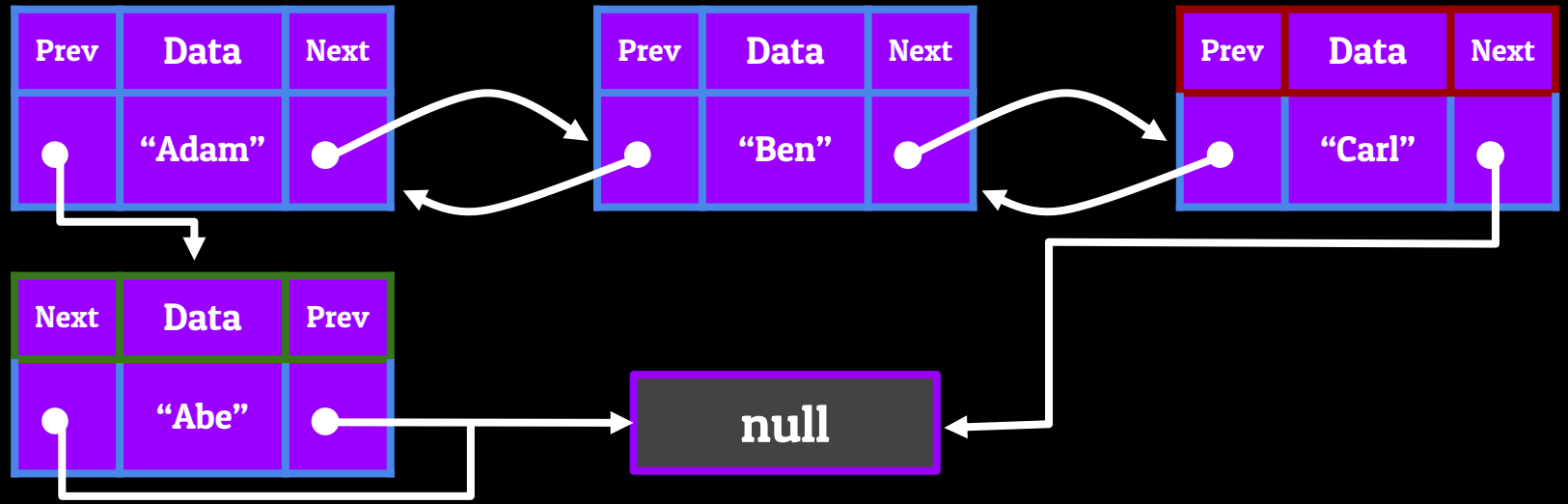


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

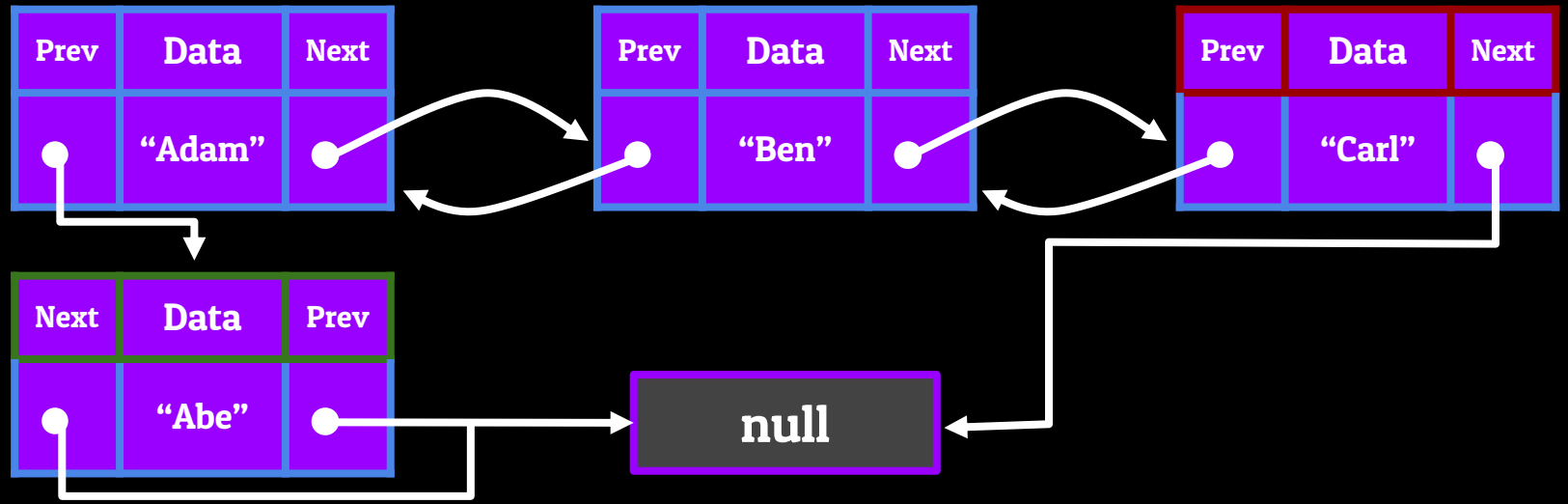


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

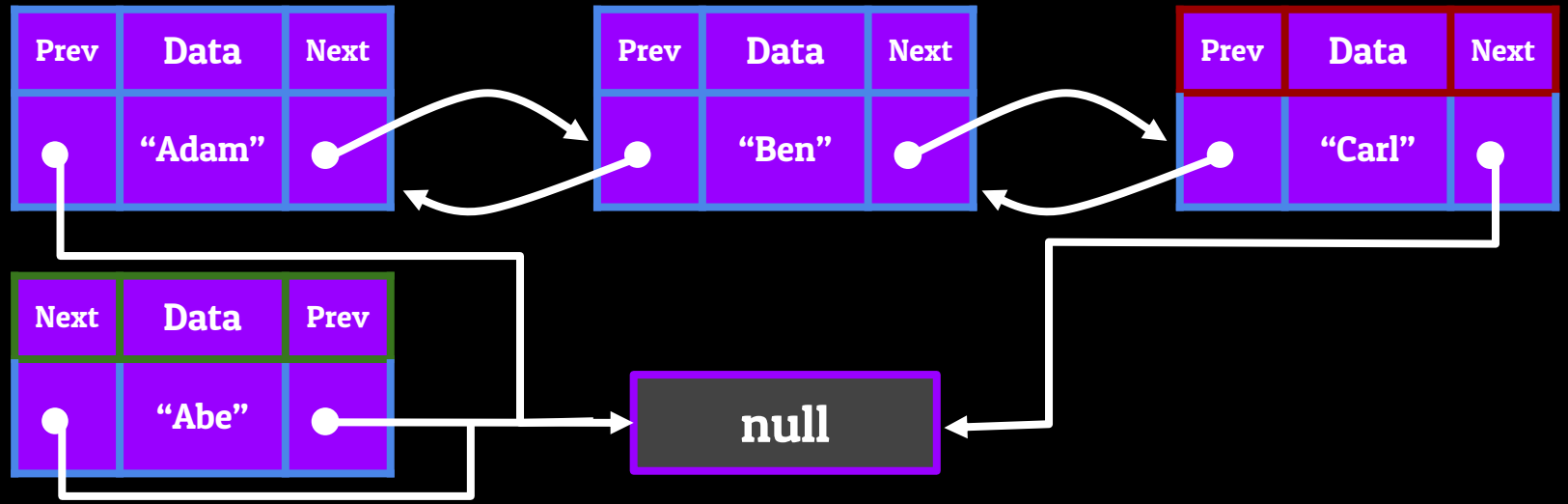


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

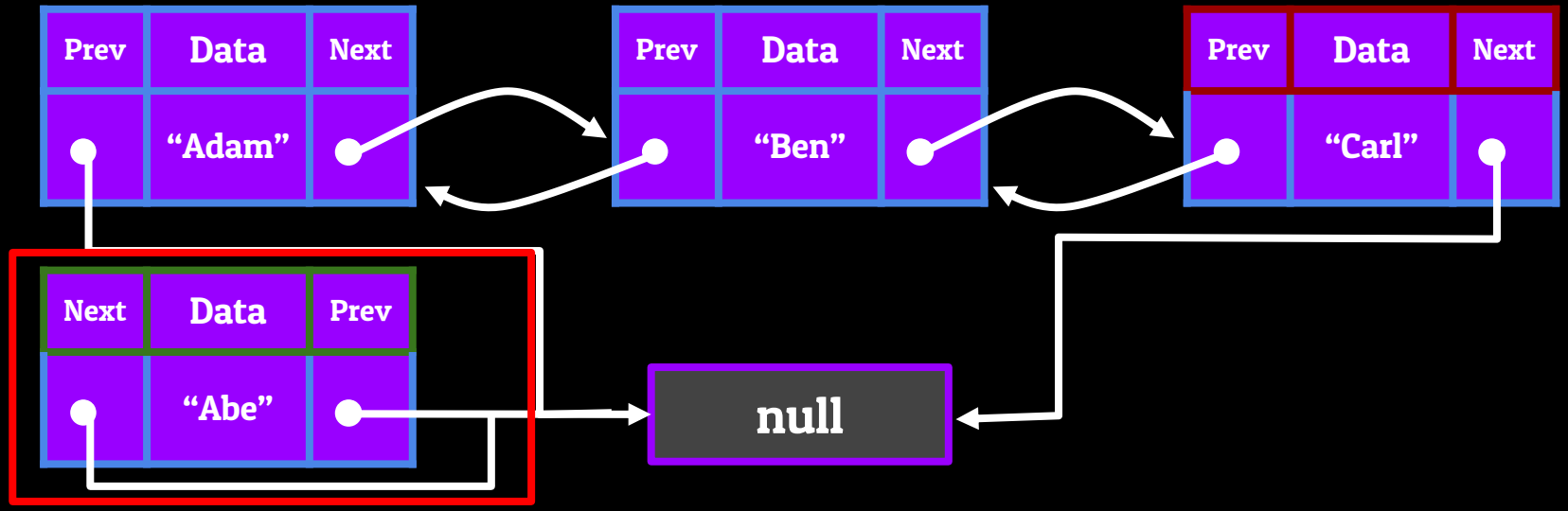


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

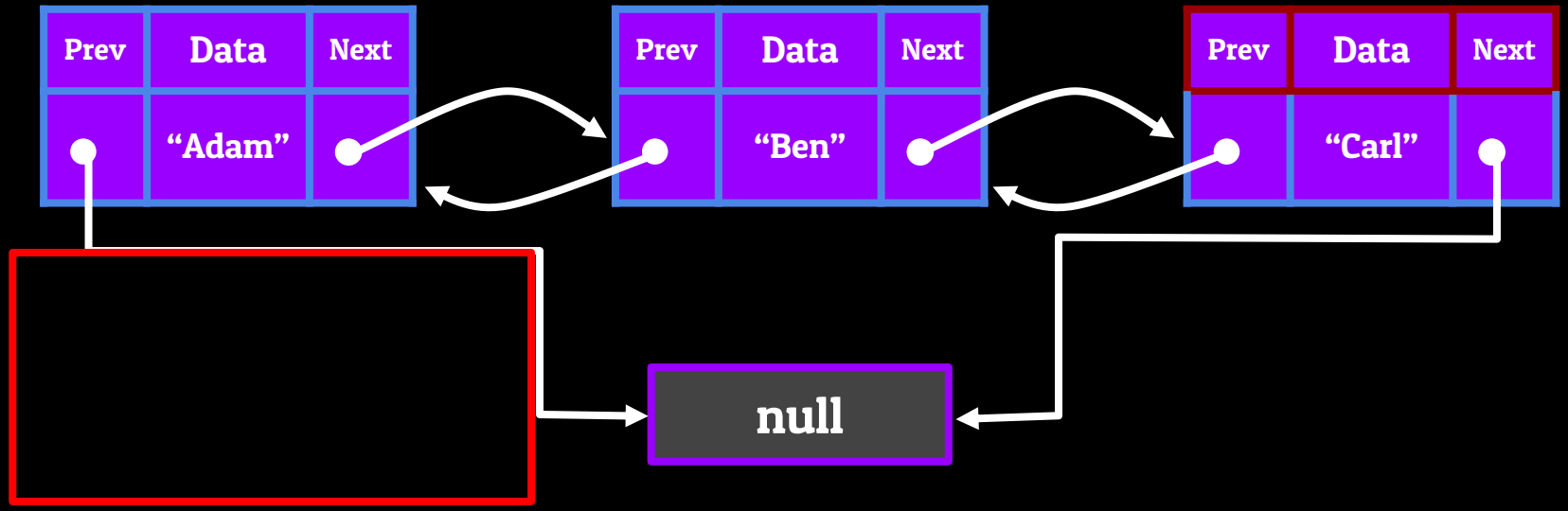


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

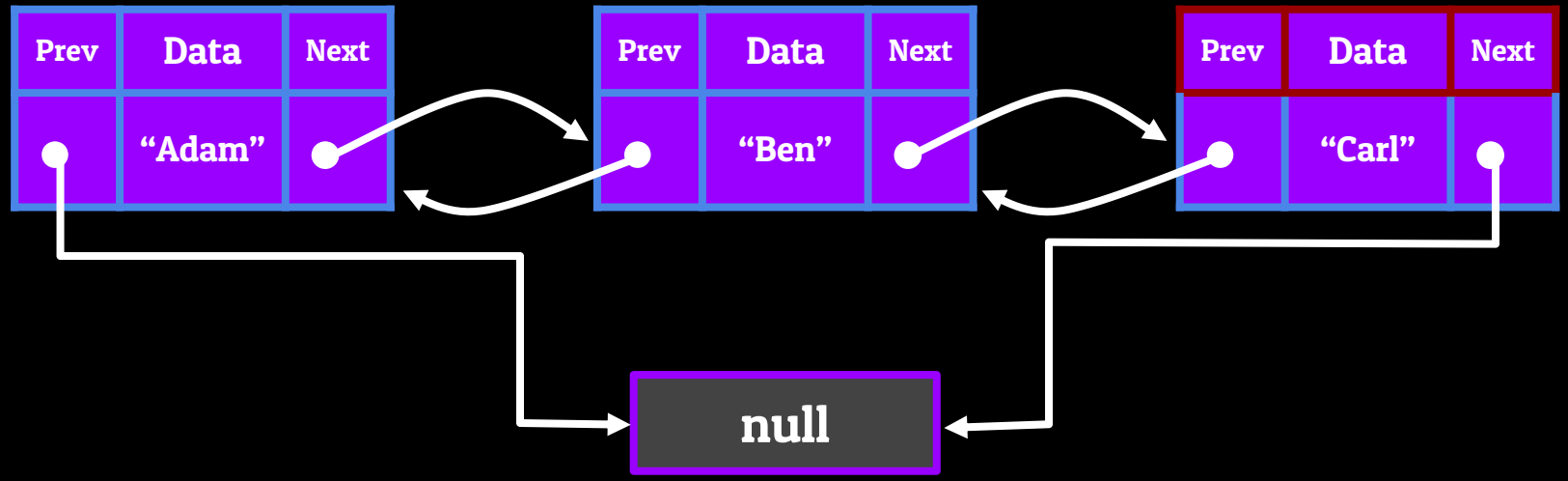


The Doubly-Linked List - Adding and Removing Information

Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

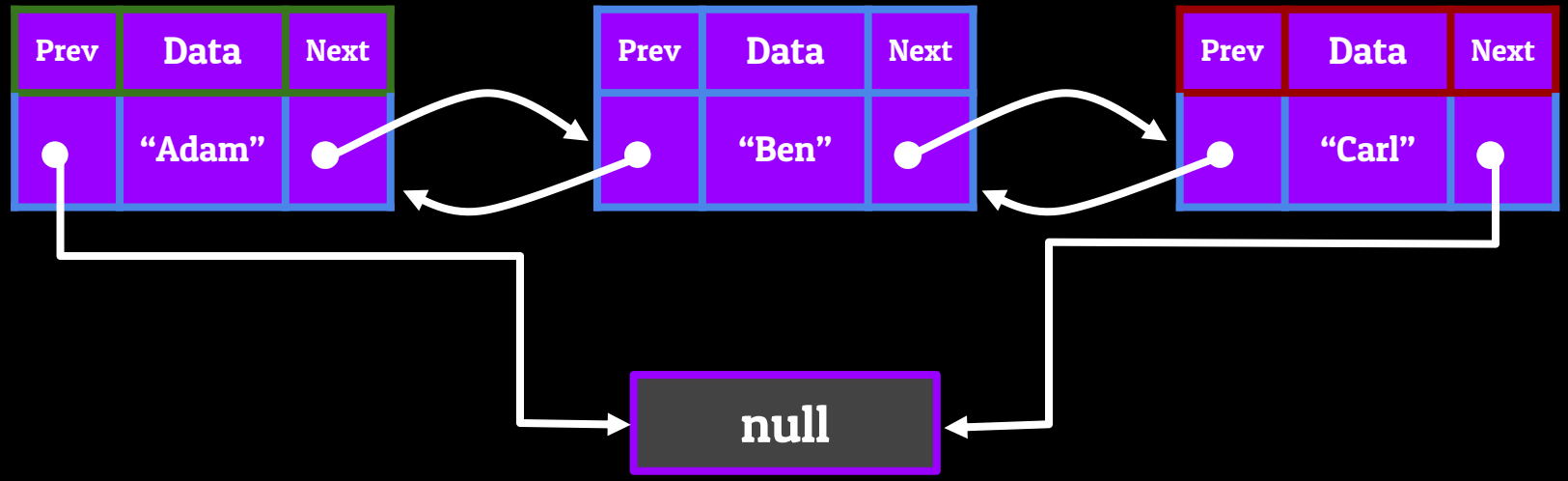


The Doubly-Linked List - Adding and Removing Information

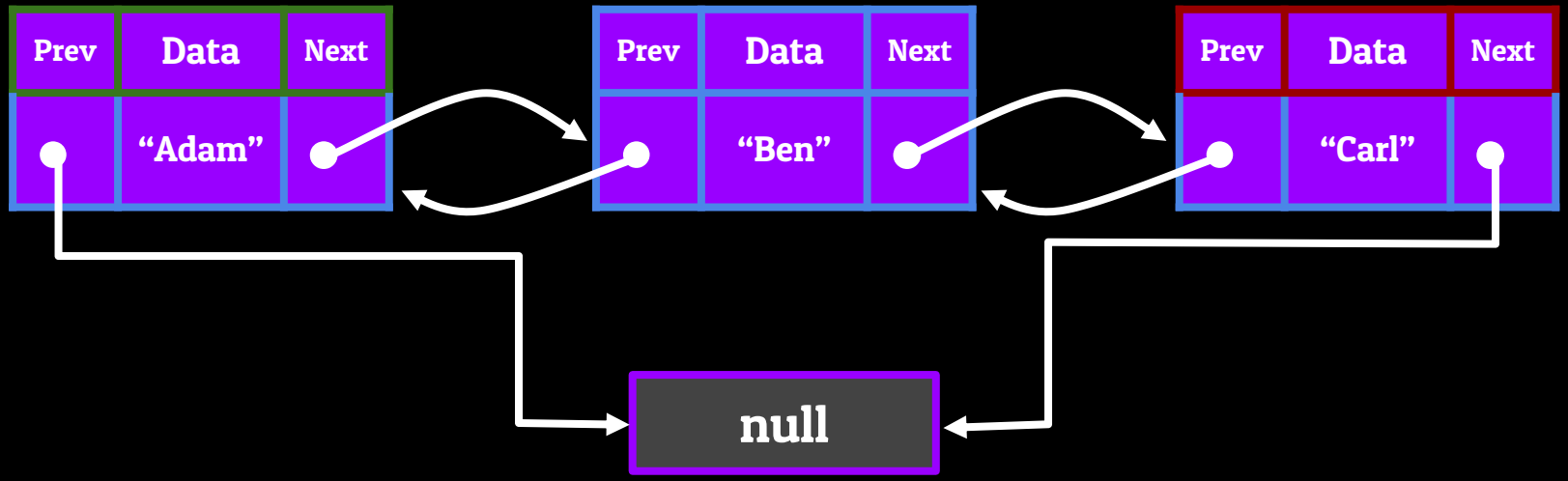
Removing from the Head of a Doubly-LinkedList

Set the head Node's next to point towards a null value

Set the second Node's previous to also point towards a null value

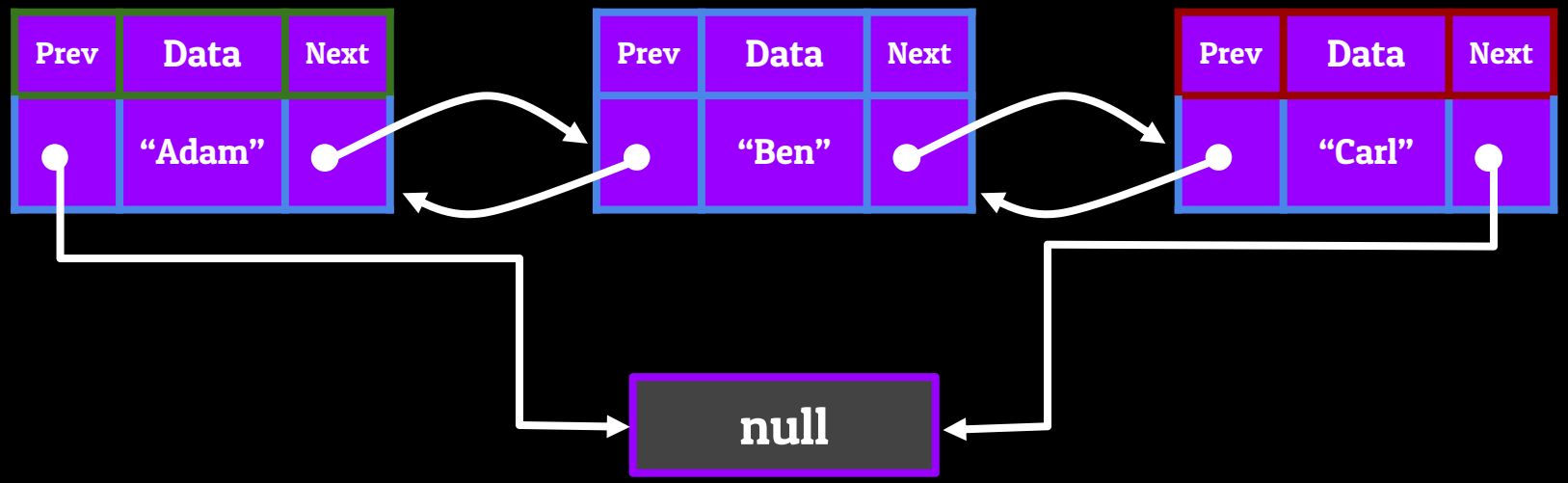


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

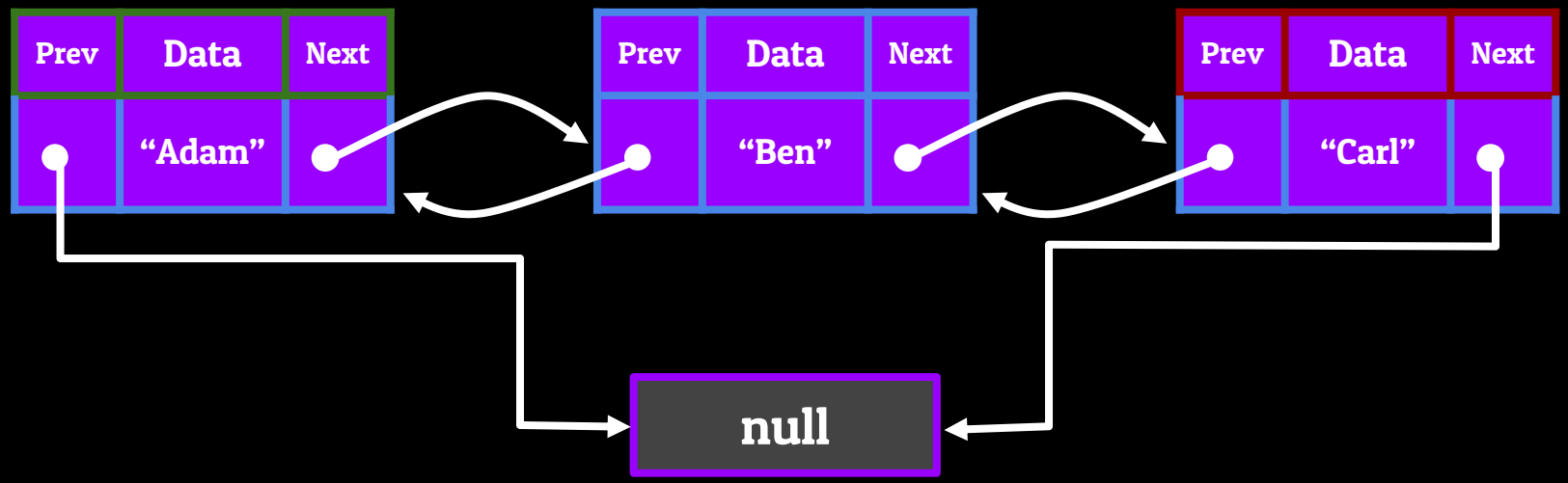
Inserting into the Middle of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

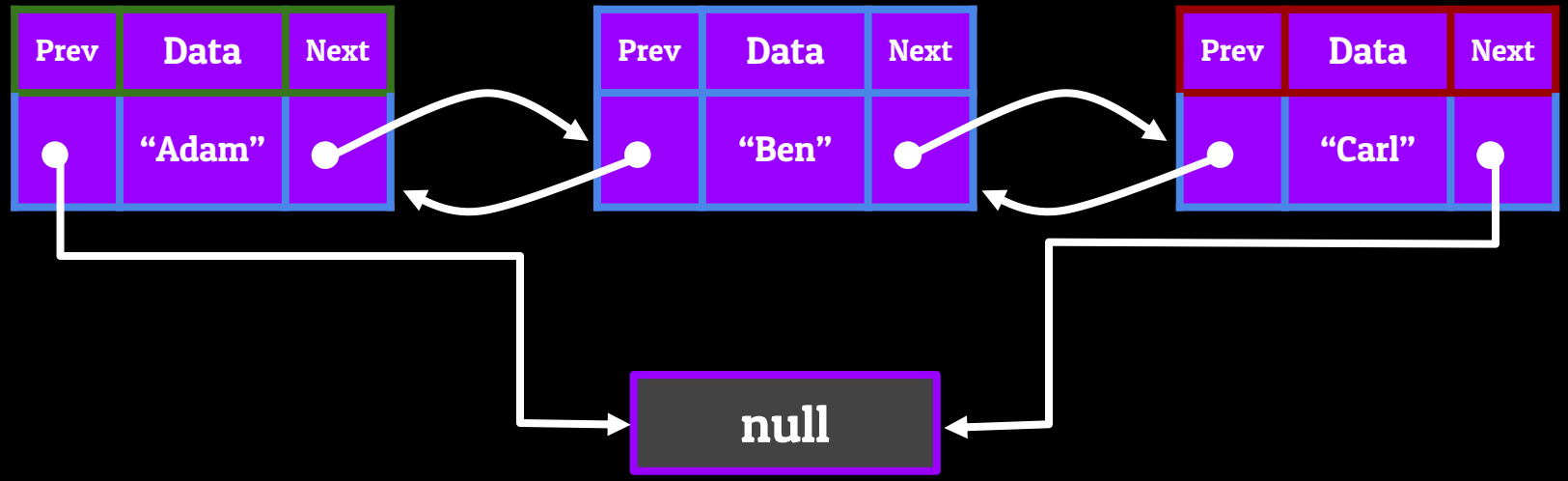


The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at



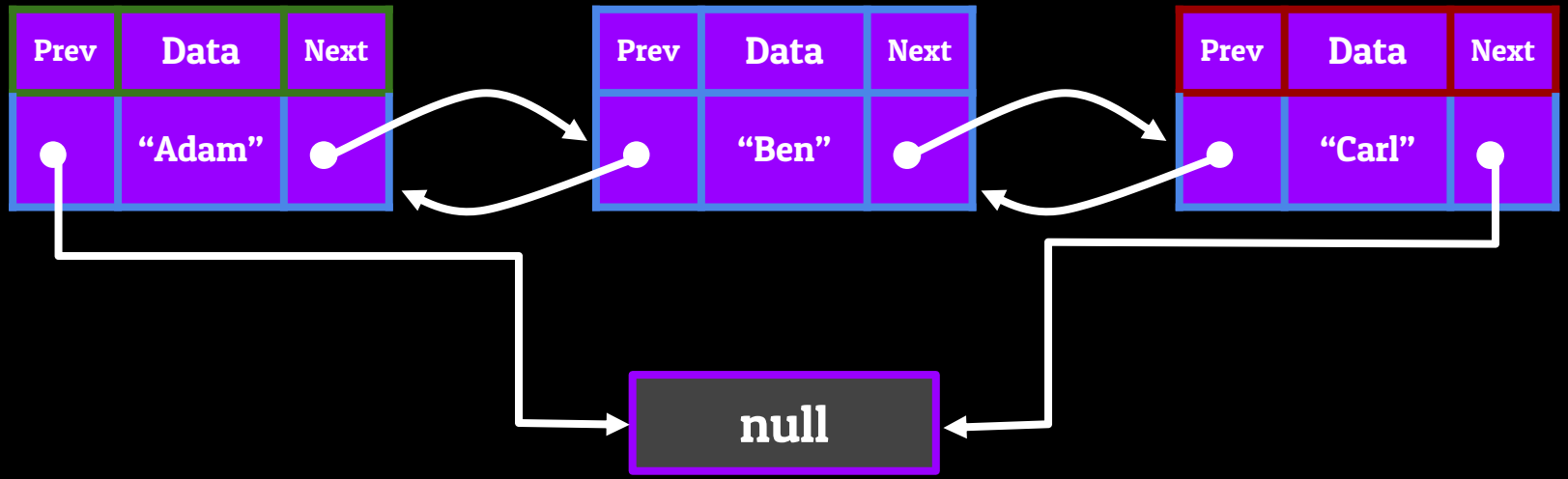
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



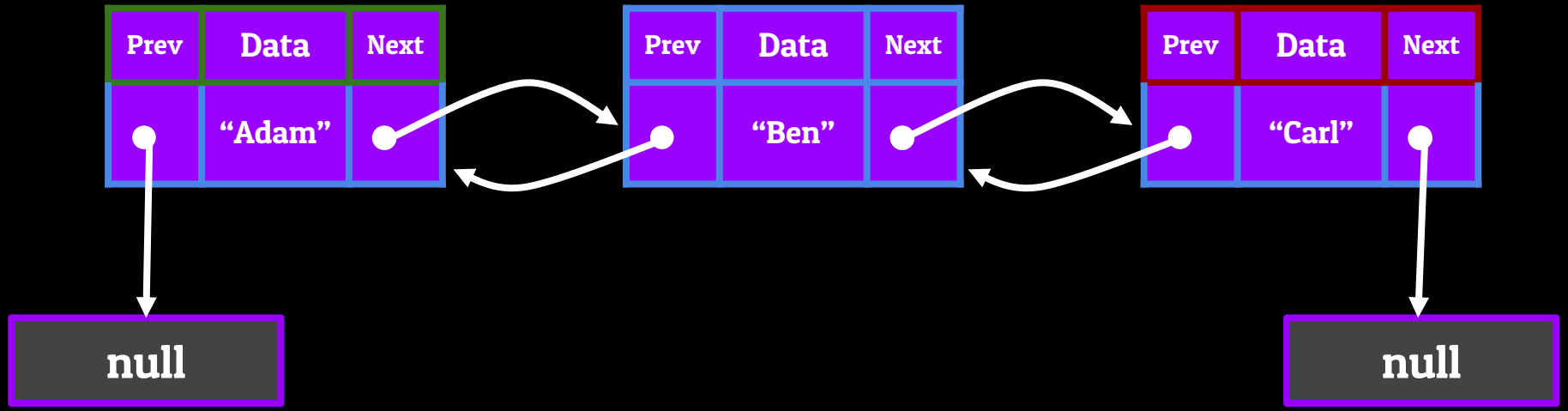
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



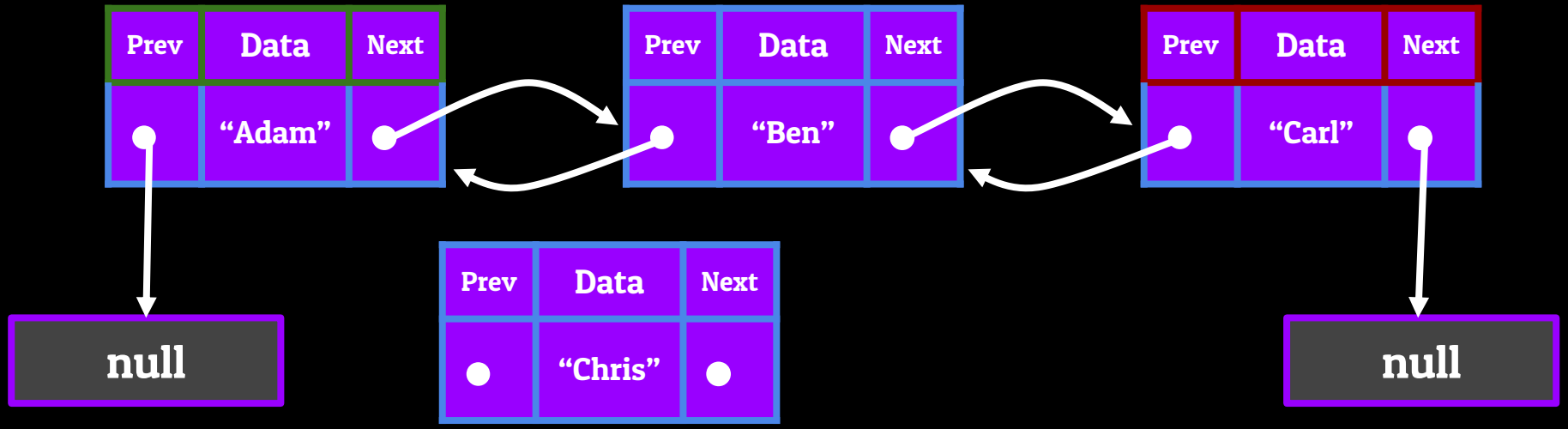
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



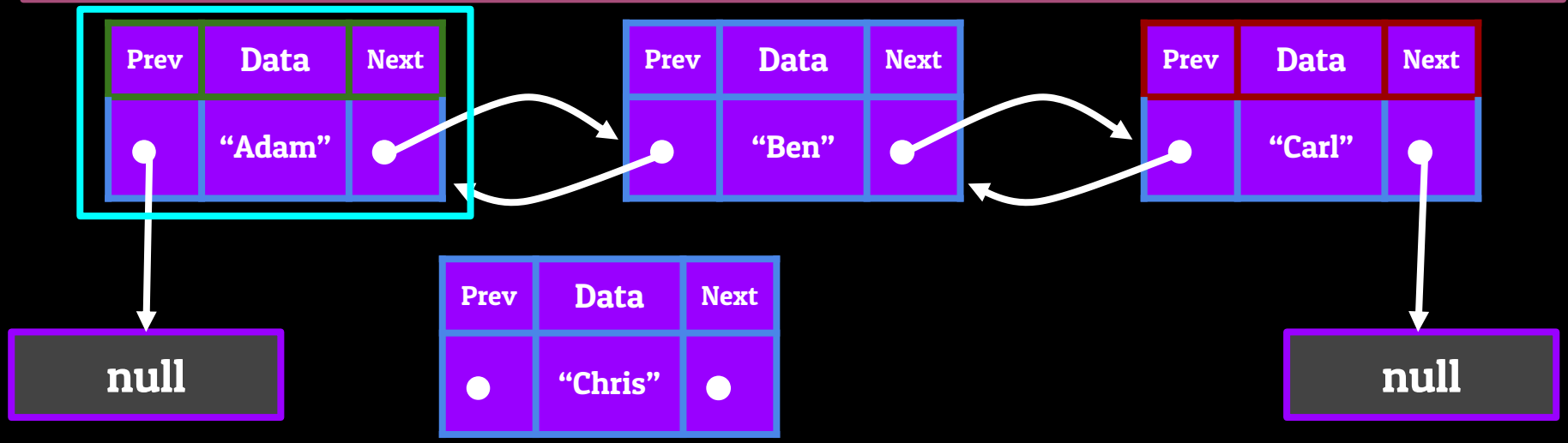
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



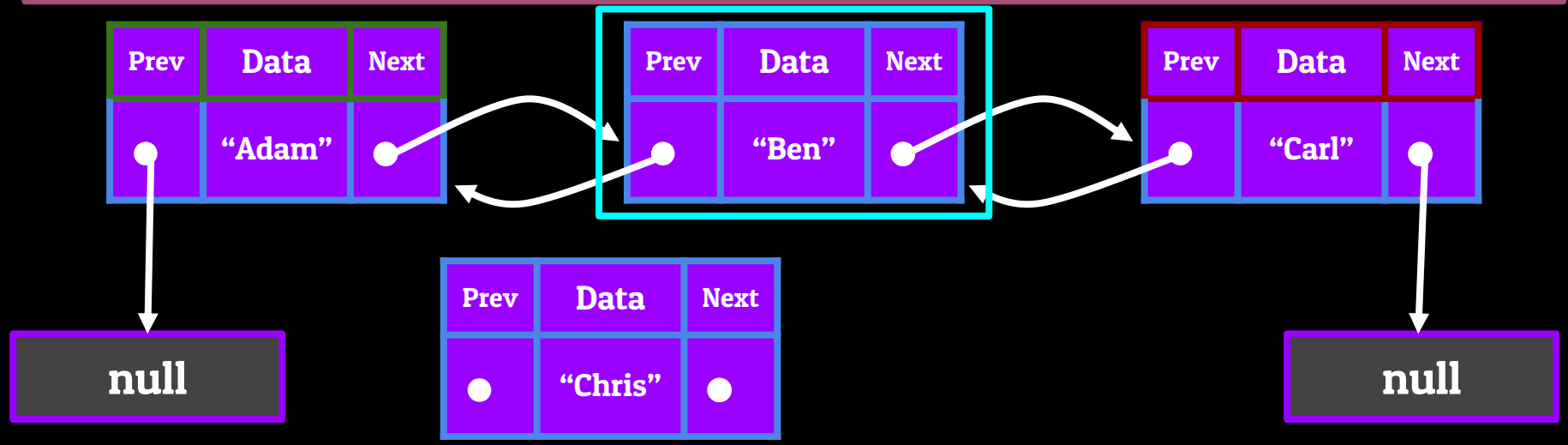
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



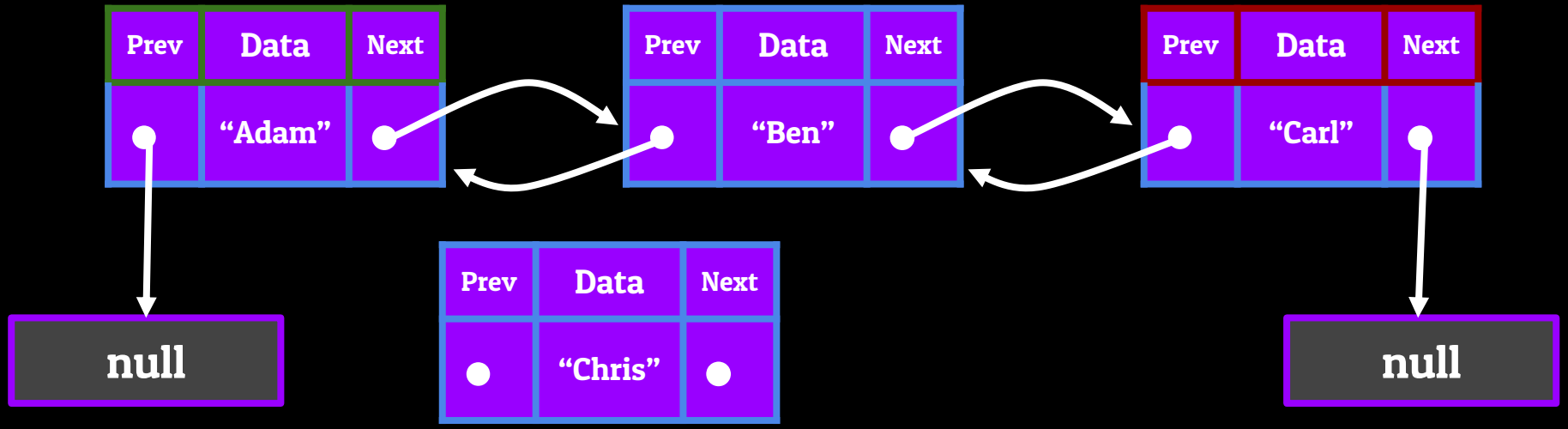
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



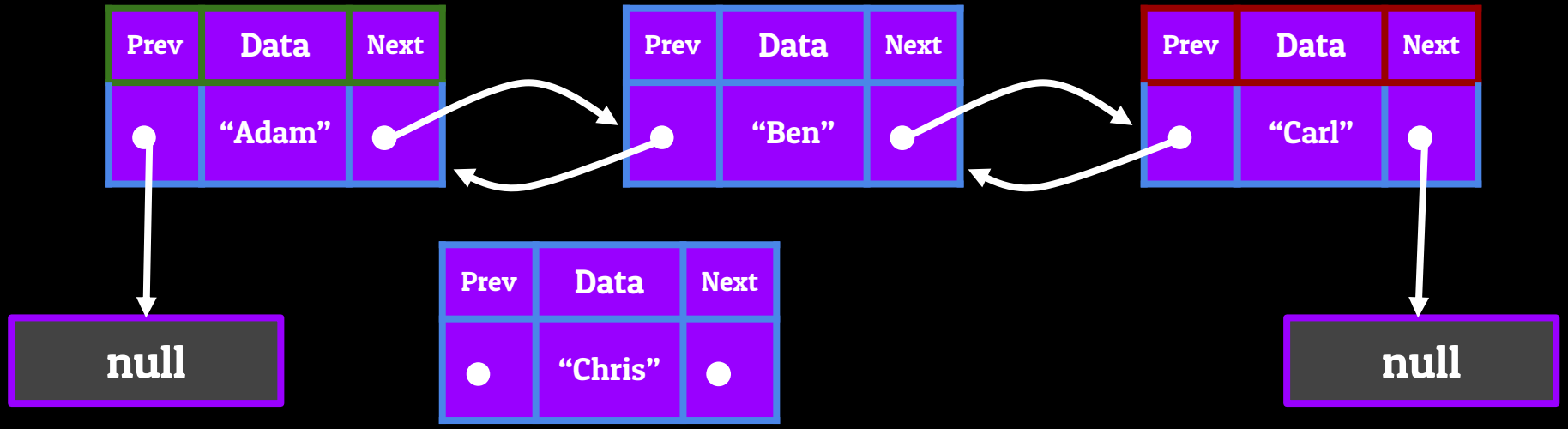
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



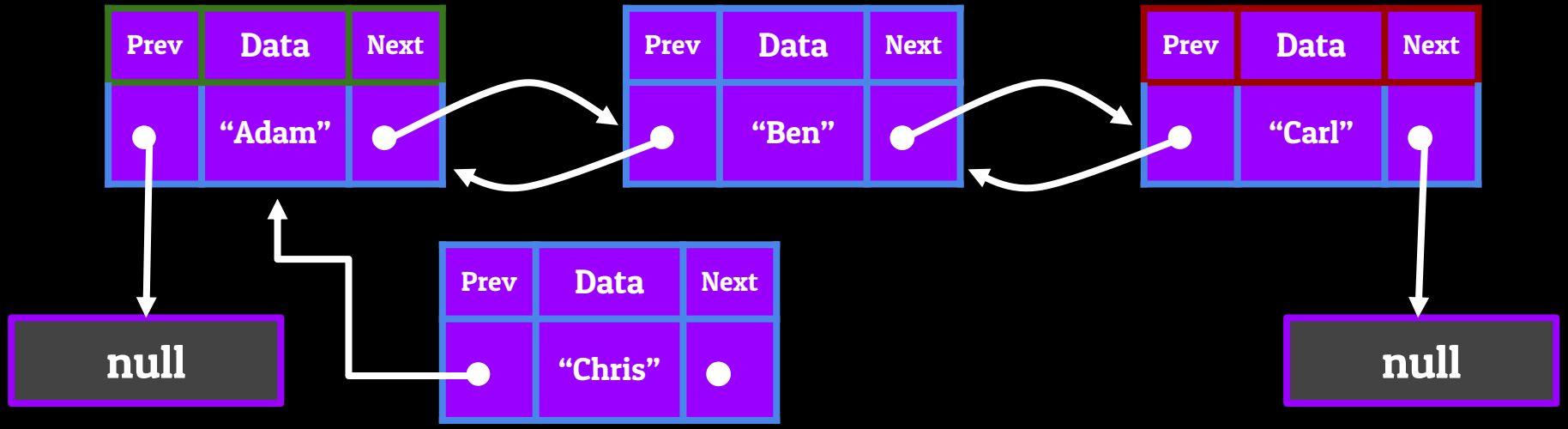
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



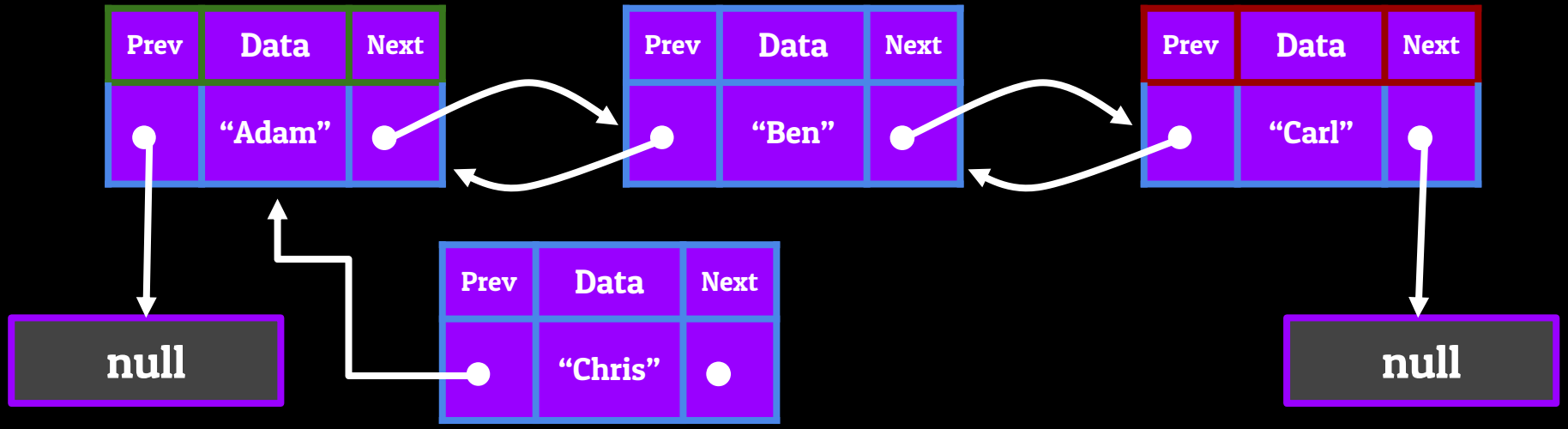
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



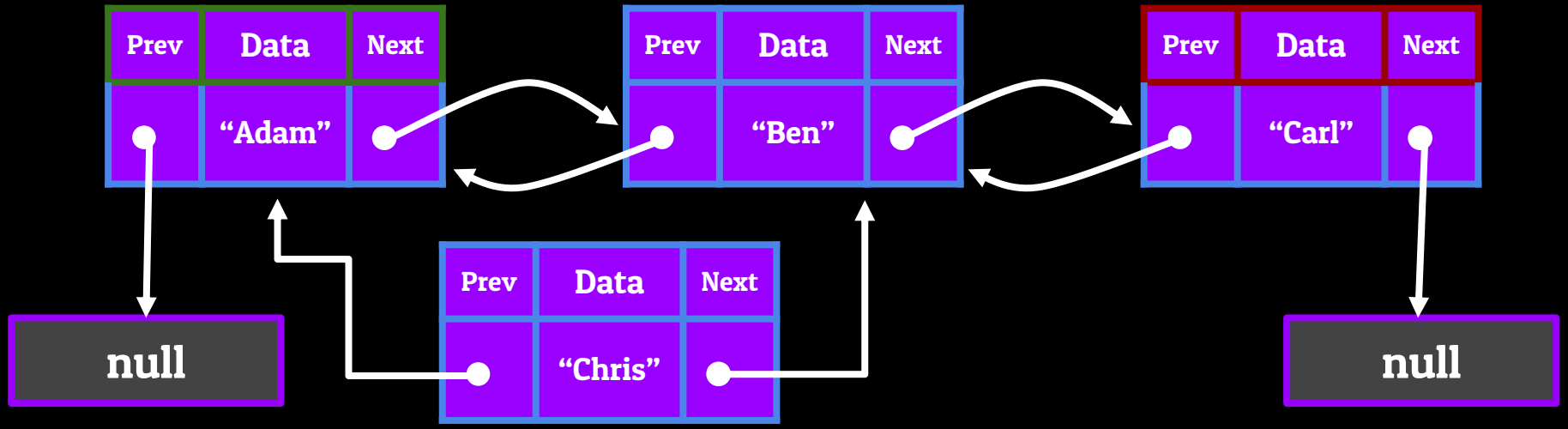
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



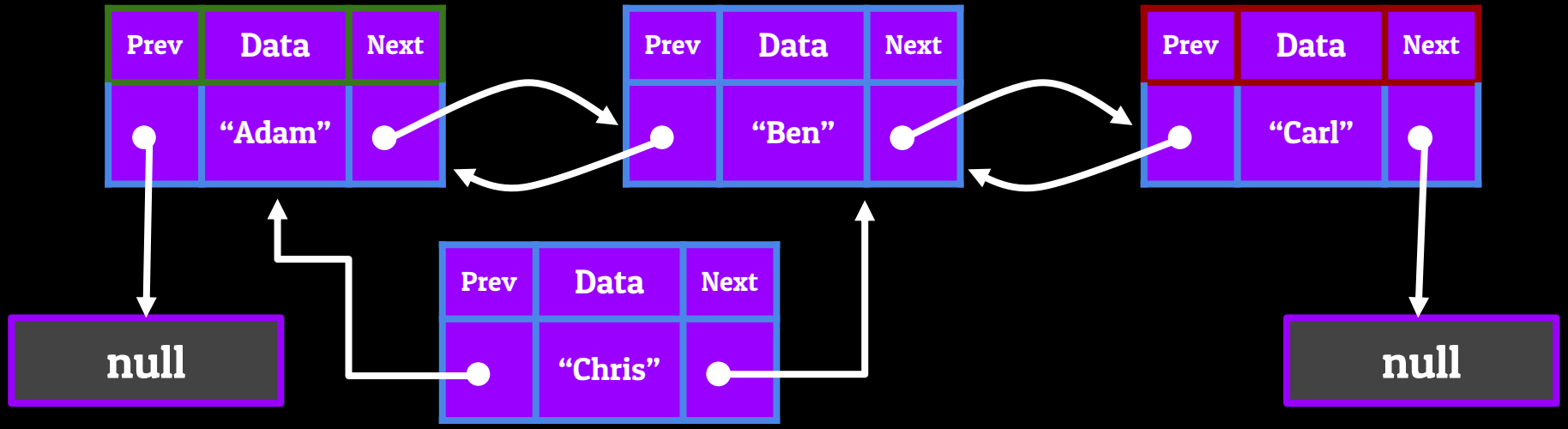
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



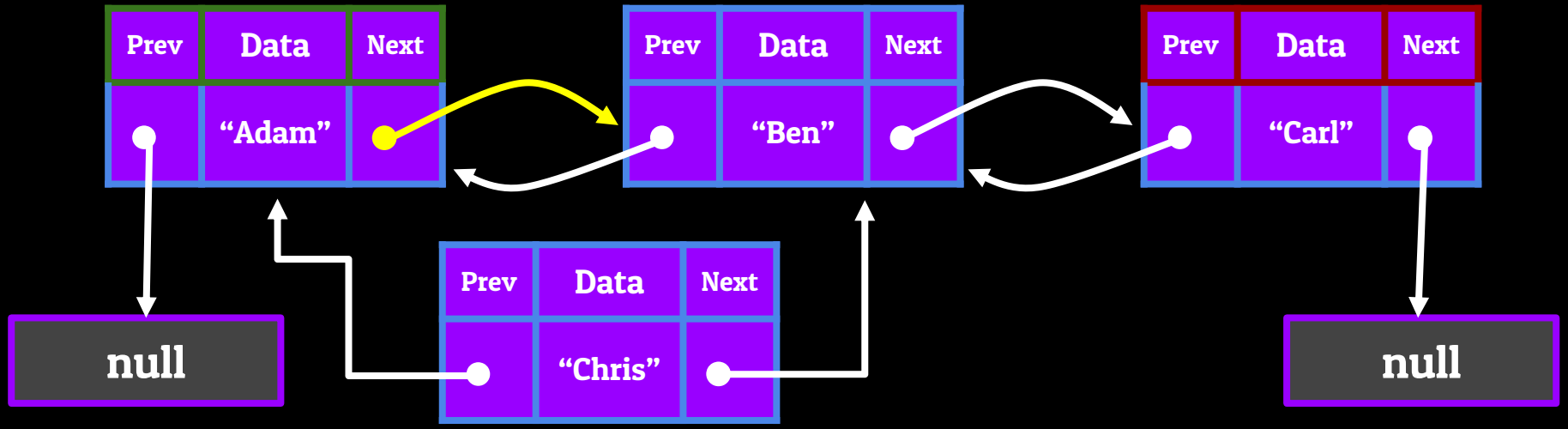
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



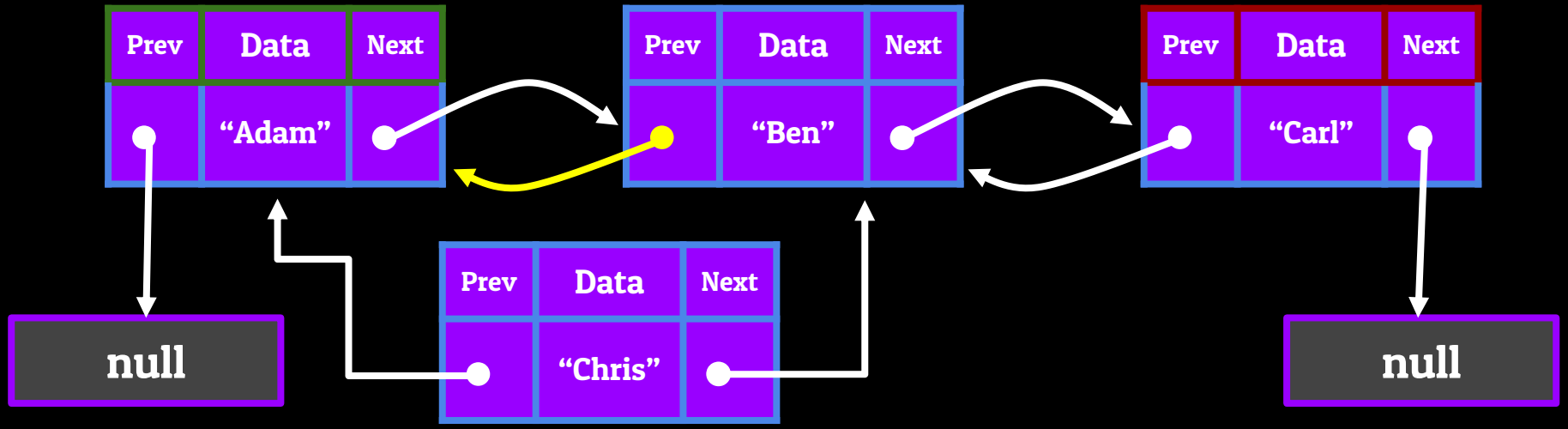
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



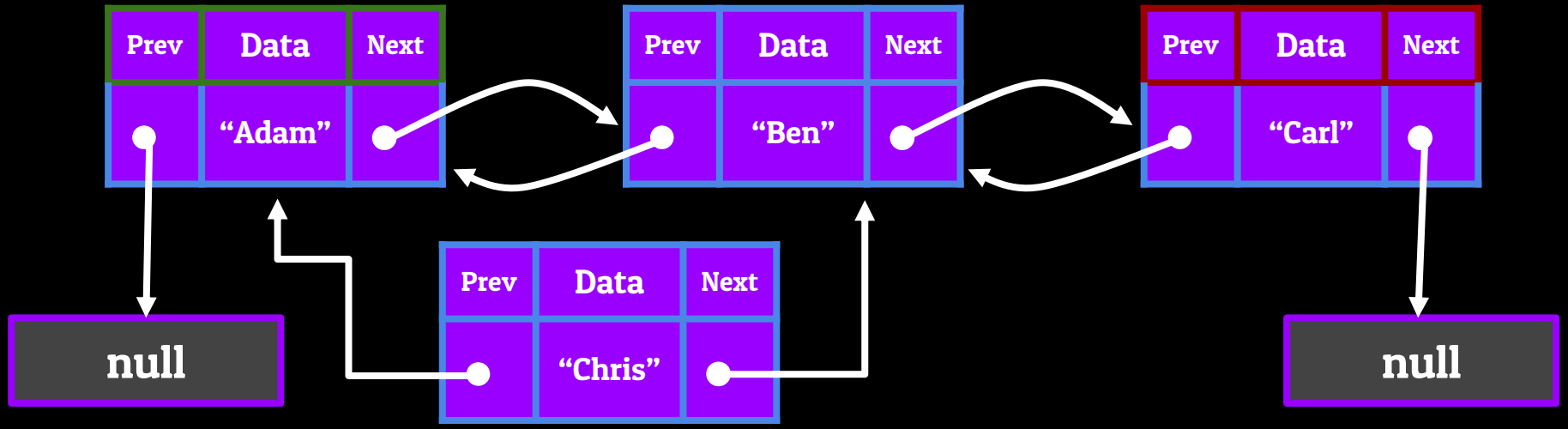
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



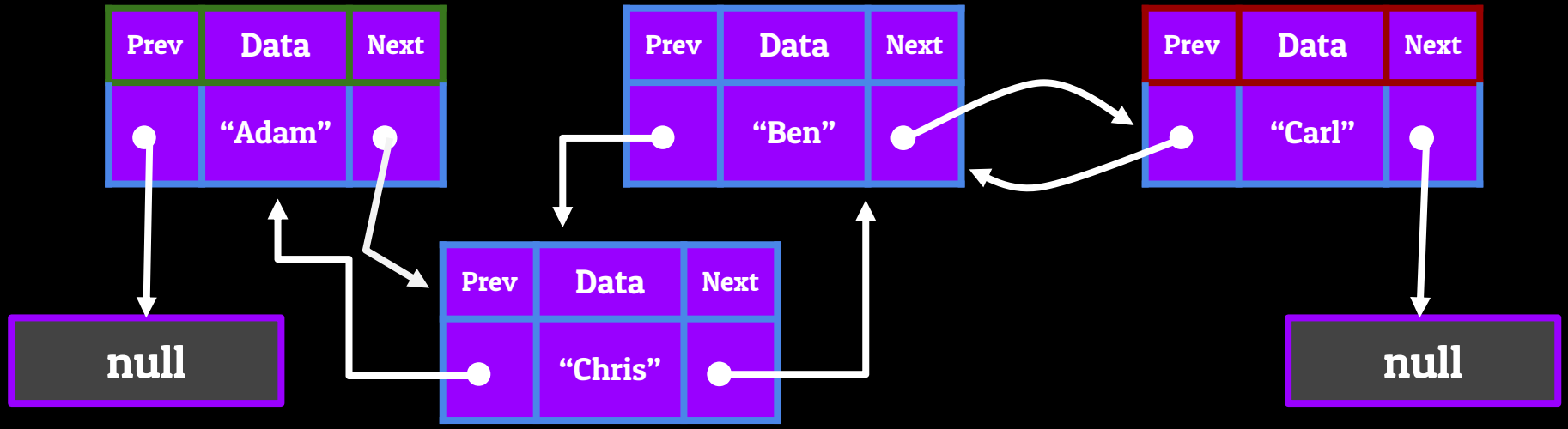
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



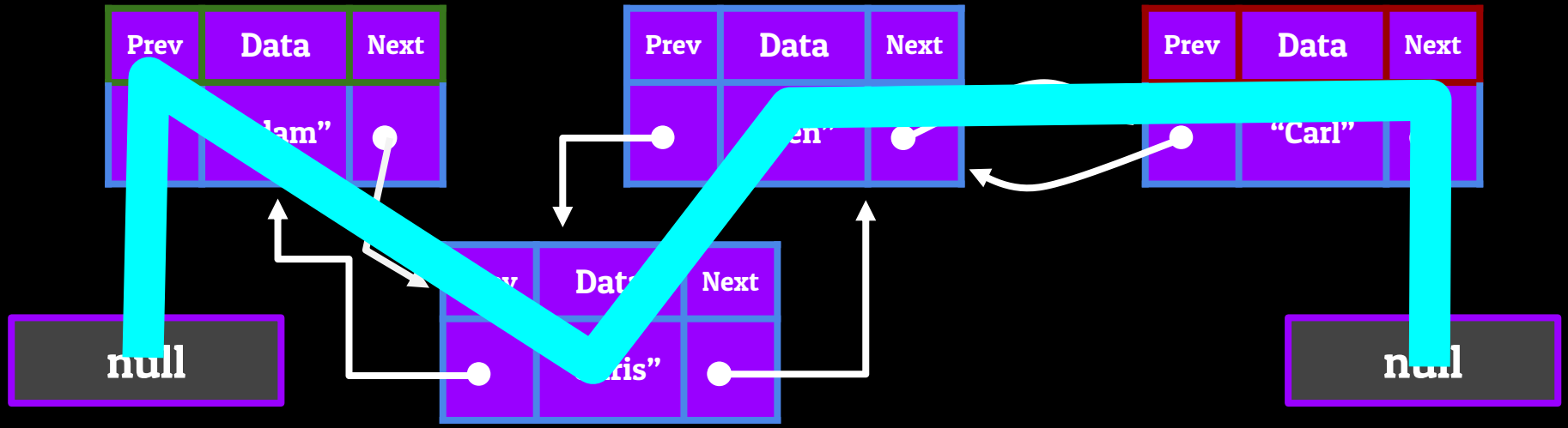
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node



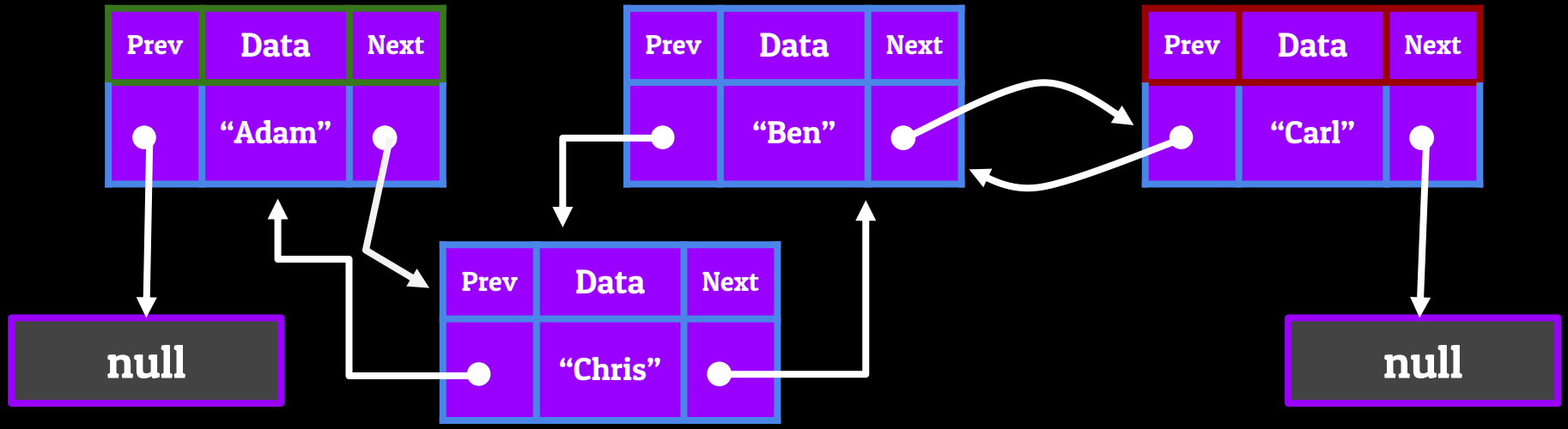
The Doubly-Linked List - Adding and Removing Information

Inserting into the Middle of a Doubly-LinkedList

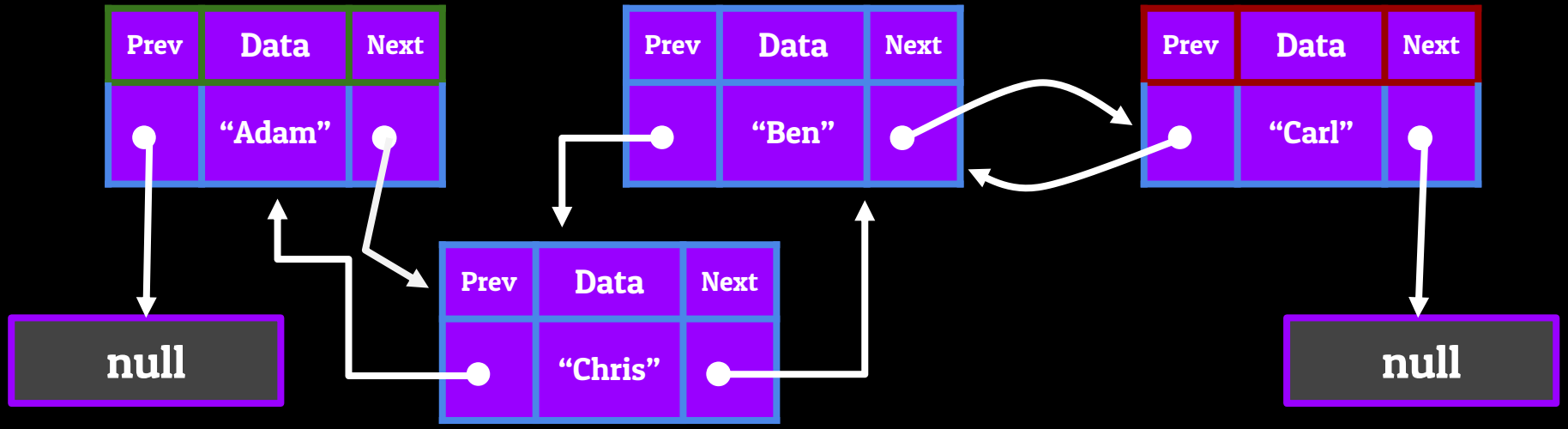
Set the new Node's previous to point towards the Node previous to the position you want to insert at

Set the new Node's next to point towards the Node after the position you want to insert at

Set the next and previous on the Node's before and after the one you're inserting to point towards the new Node

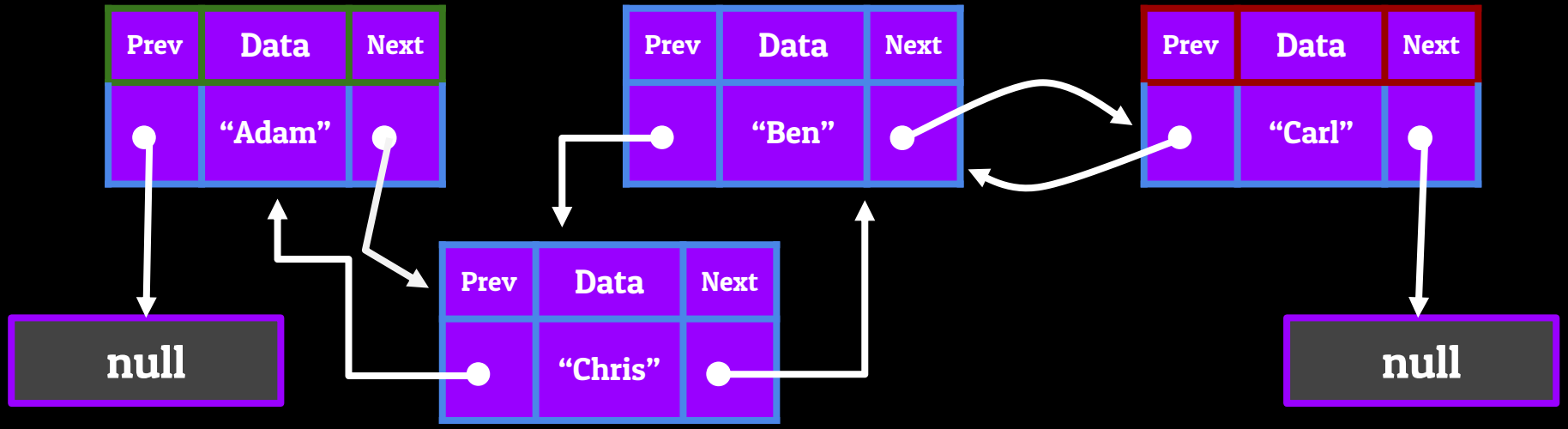


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

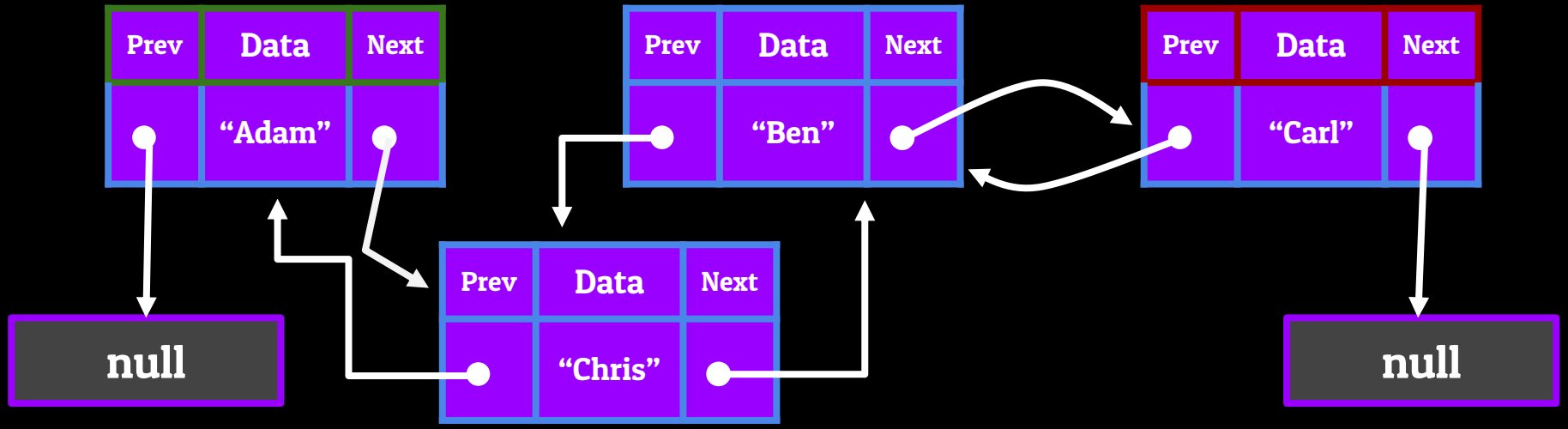
Removing from the Middle of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

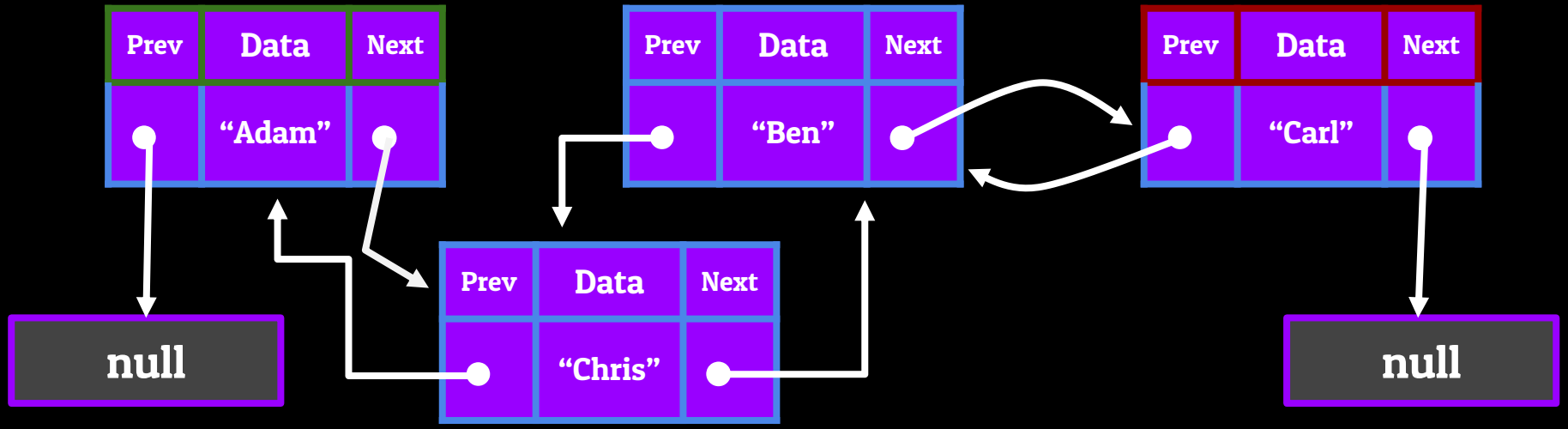


The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove



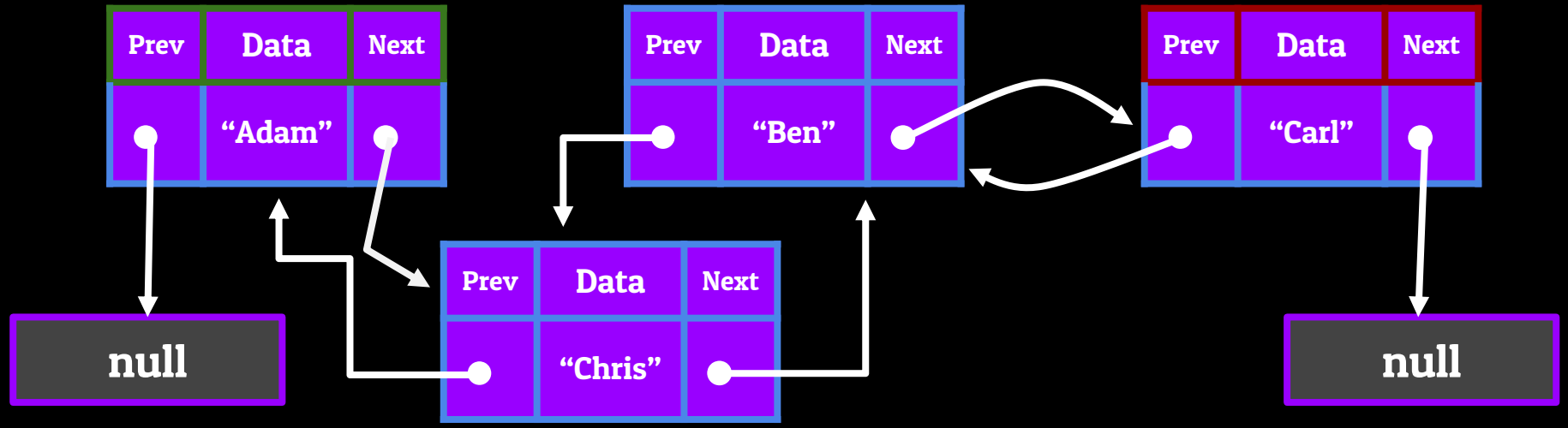
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



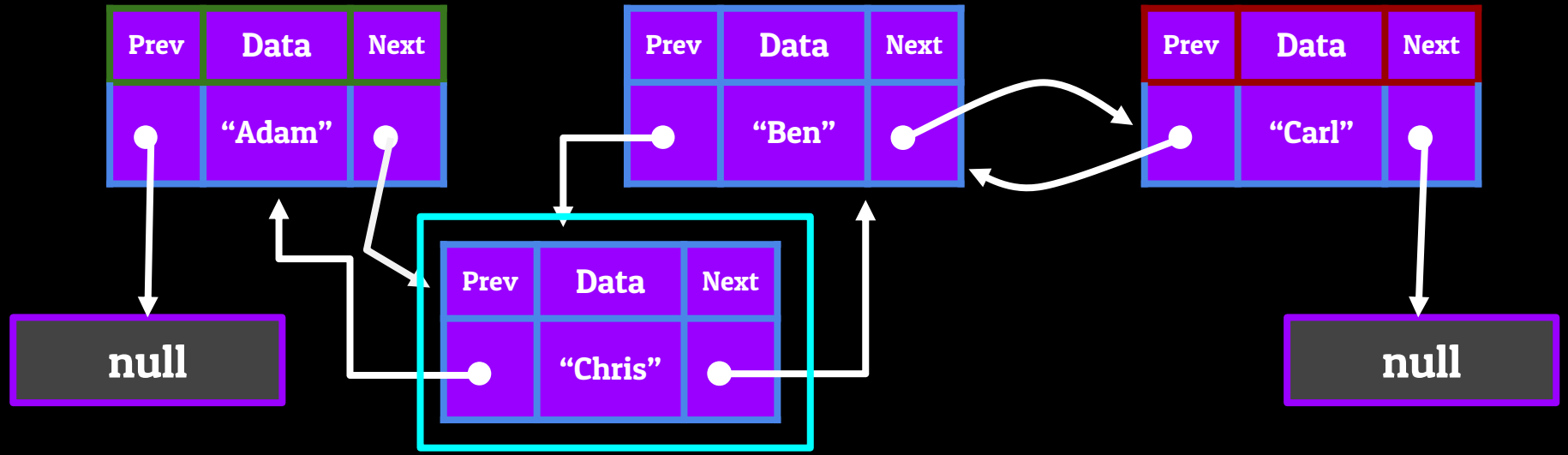
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



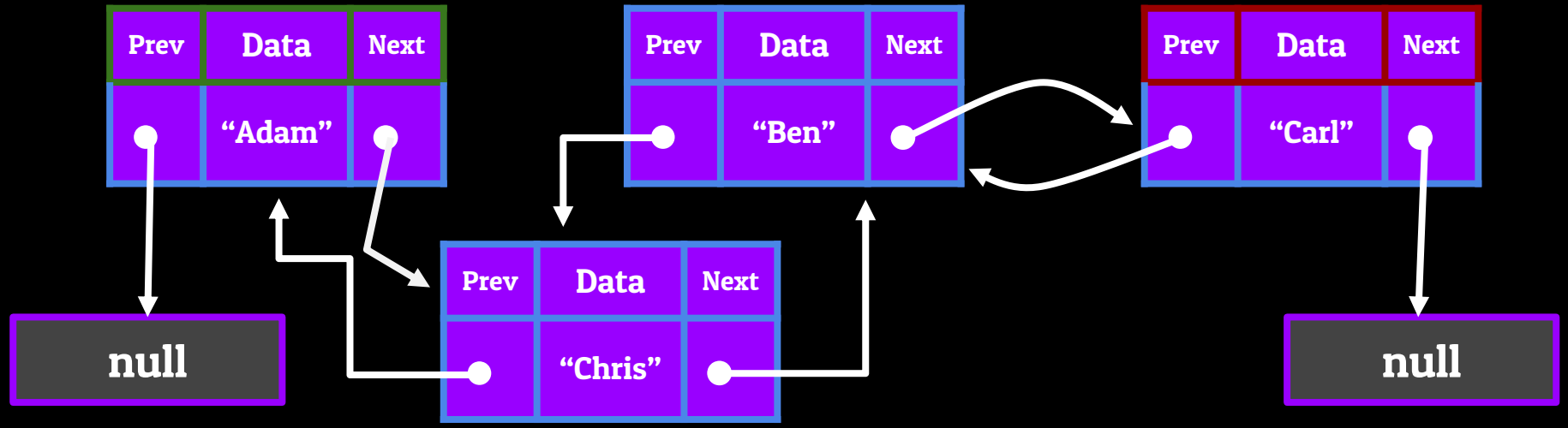
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



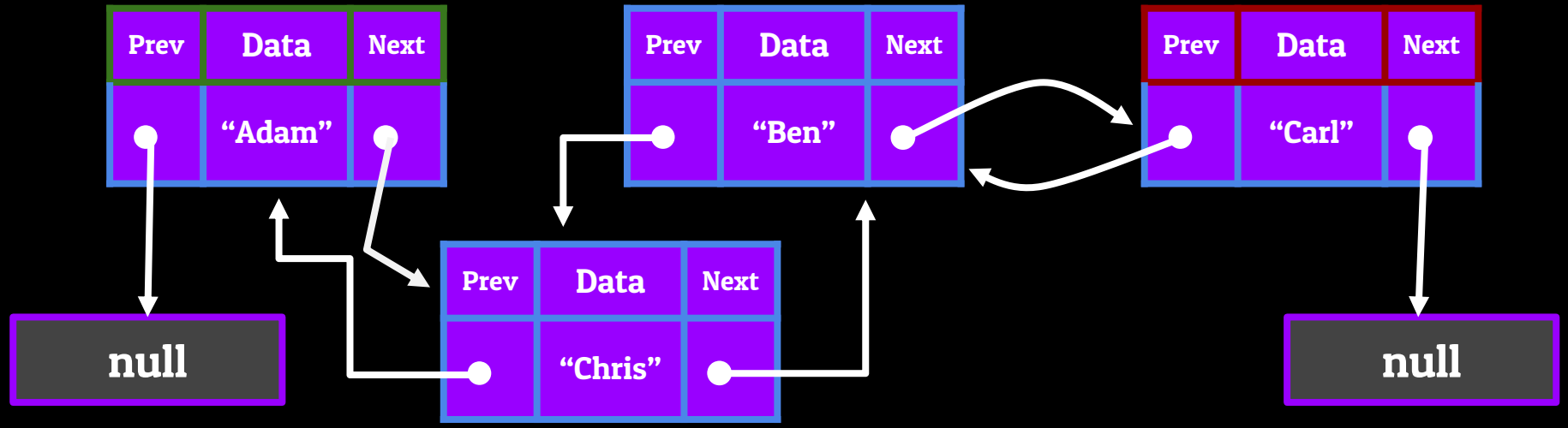
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



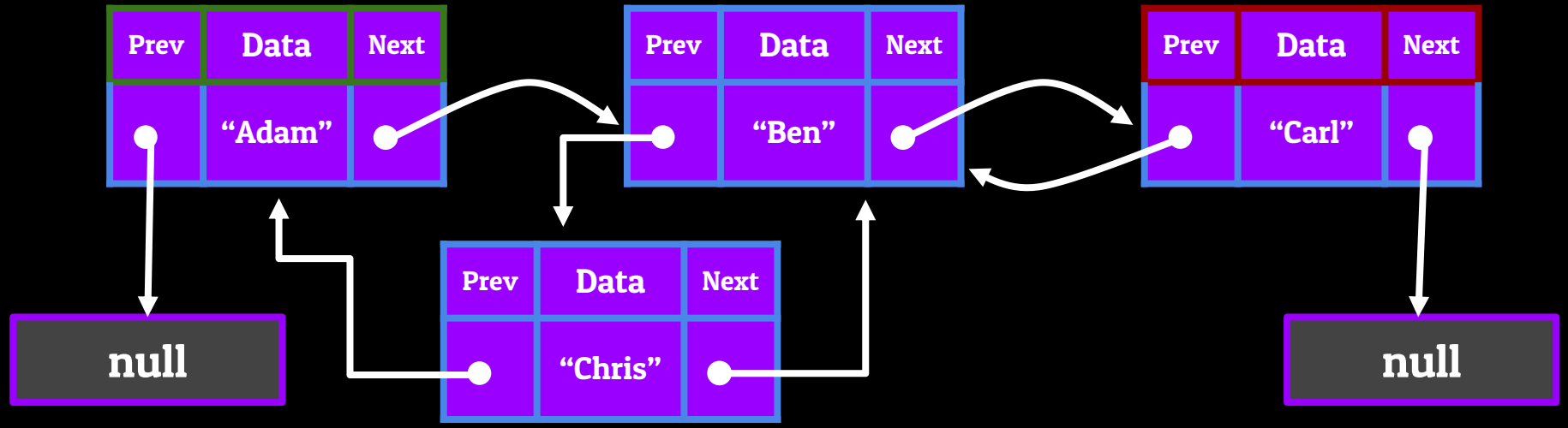
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



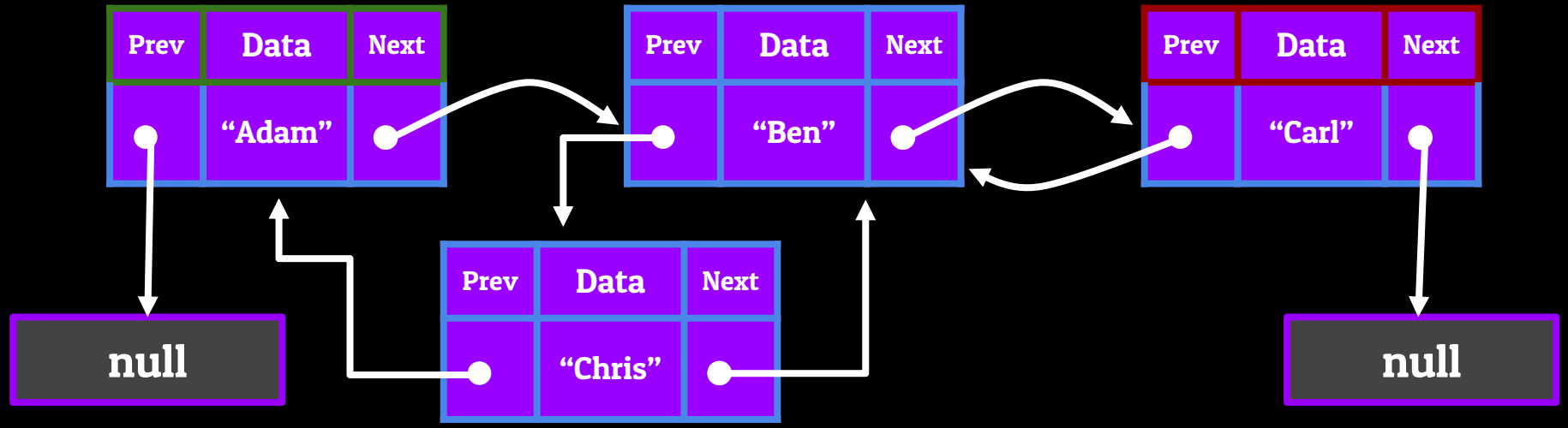
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



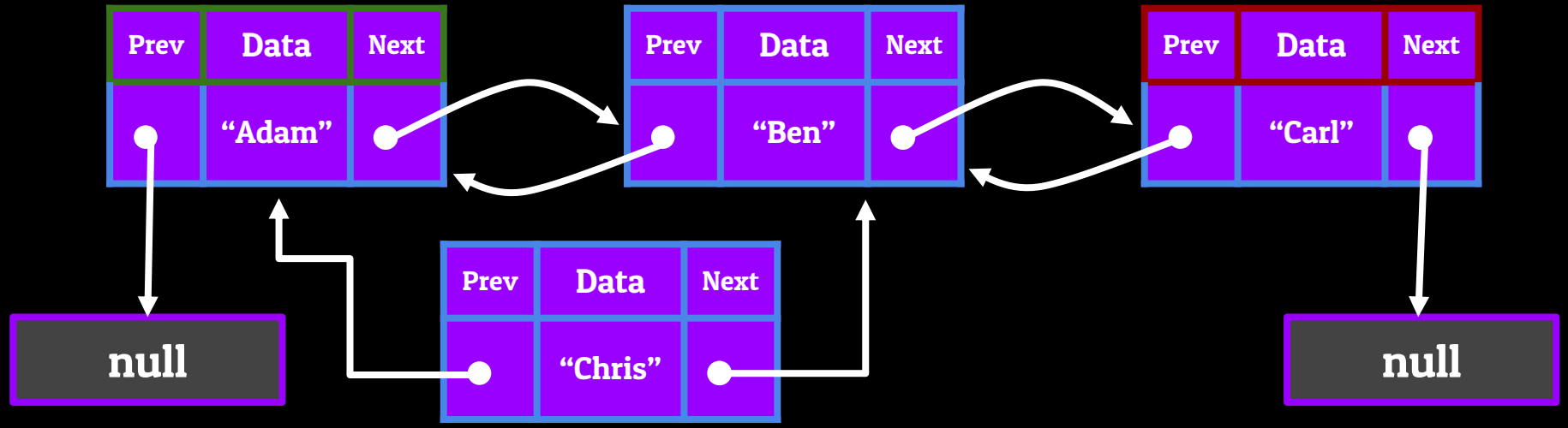
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



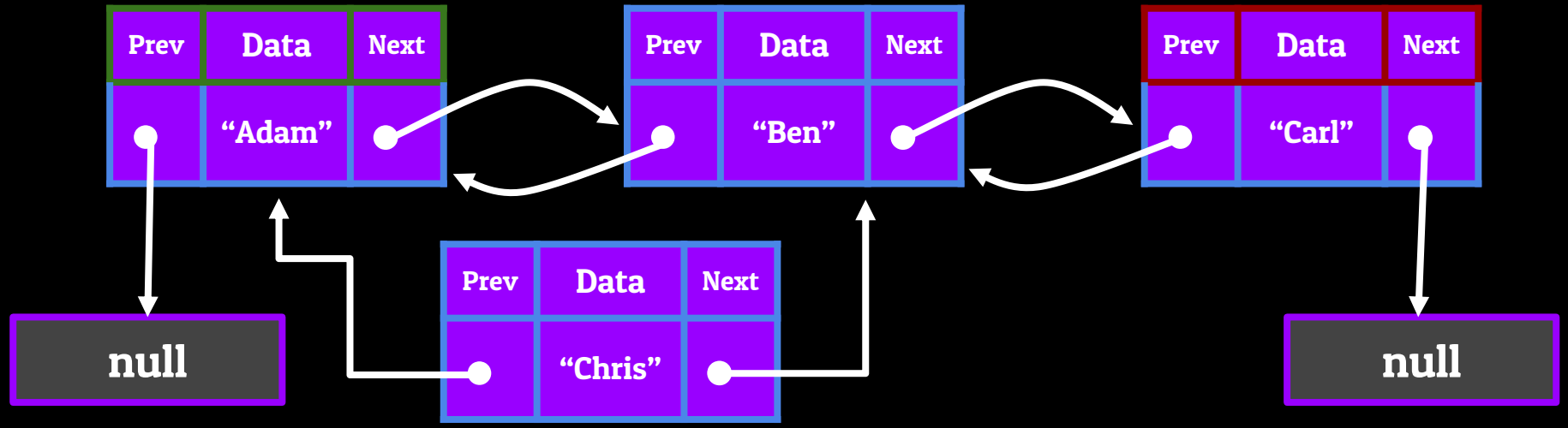
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



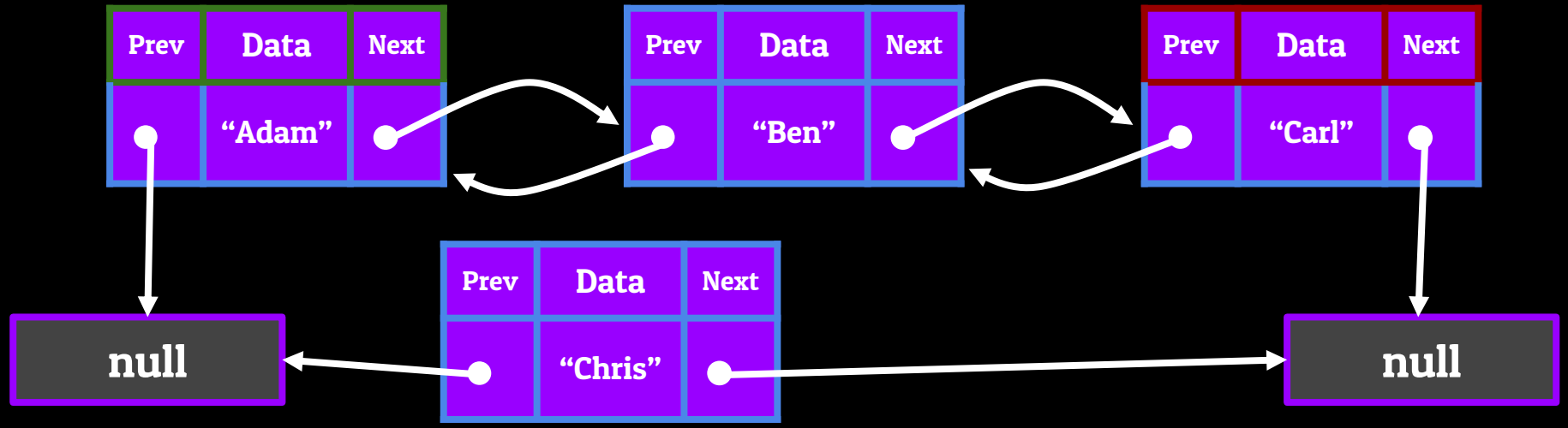
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value



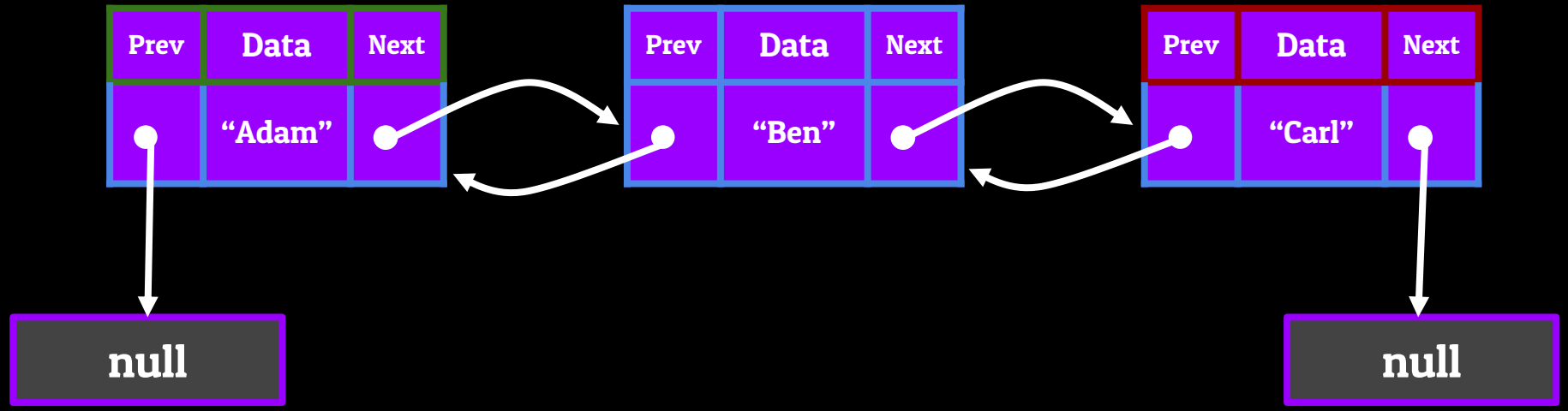
The Doubly-Linked List - Adding and Removing Information

Removing from the Middle of a Doubly-LinkedList

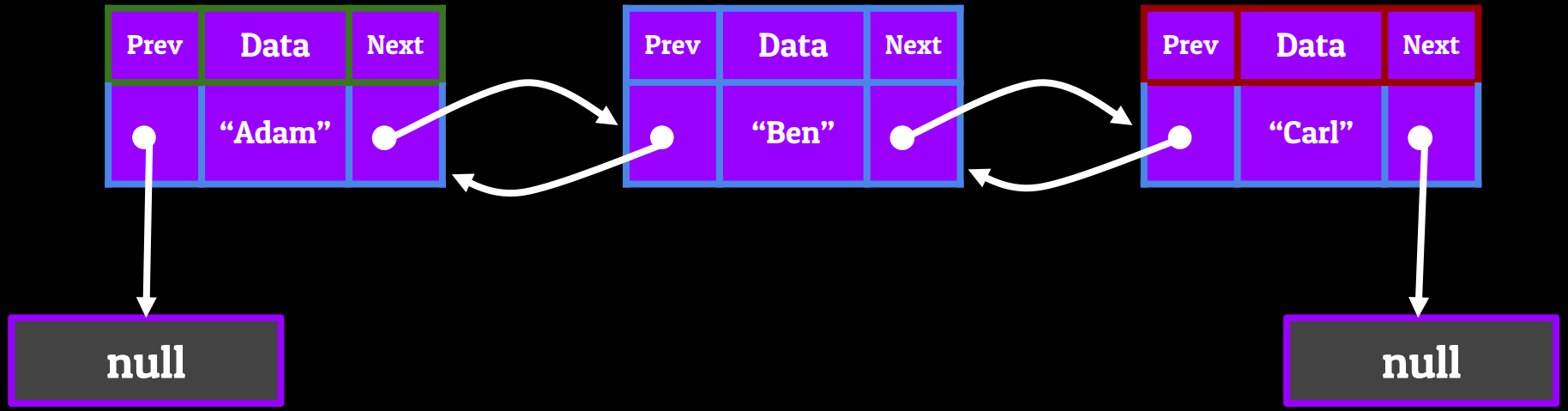
Set the Node before the one we want to remove's next to point towards the Node after the one we want to remove

Set the Node after the one we want to remove's previous to point towards the Node before the one we want to remove

Set both pointers of the Node we want to remove to point towards a null value

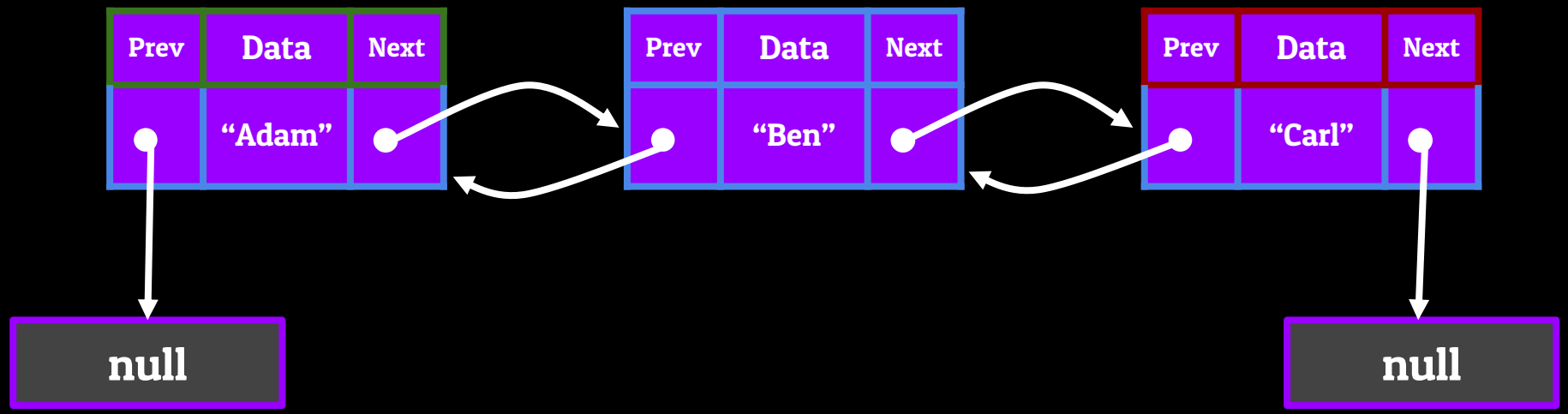


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

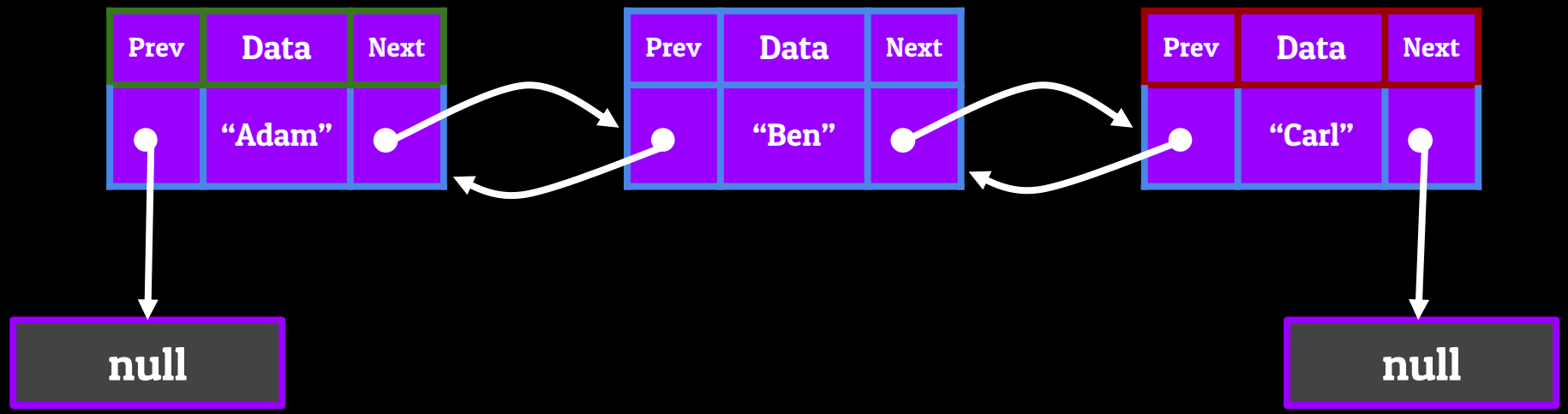
Add to the Tail of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

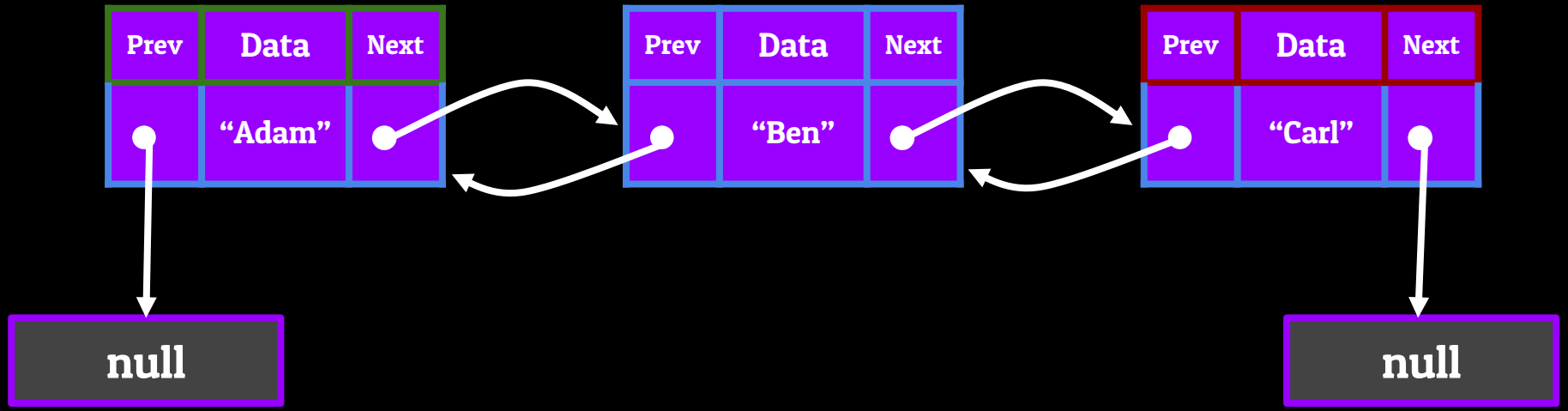


The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List



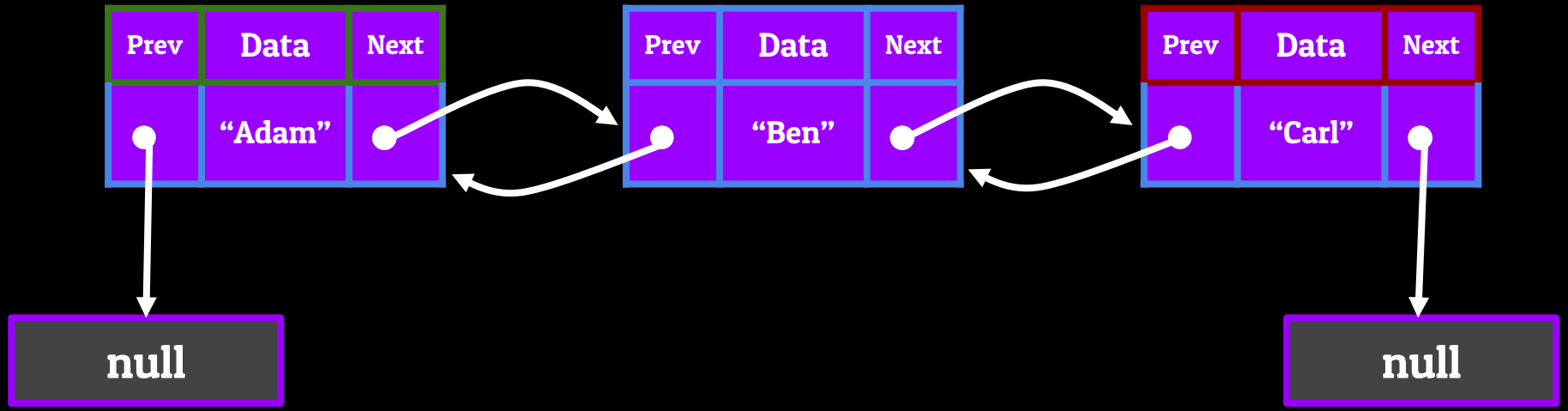
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



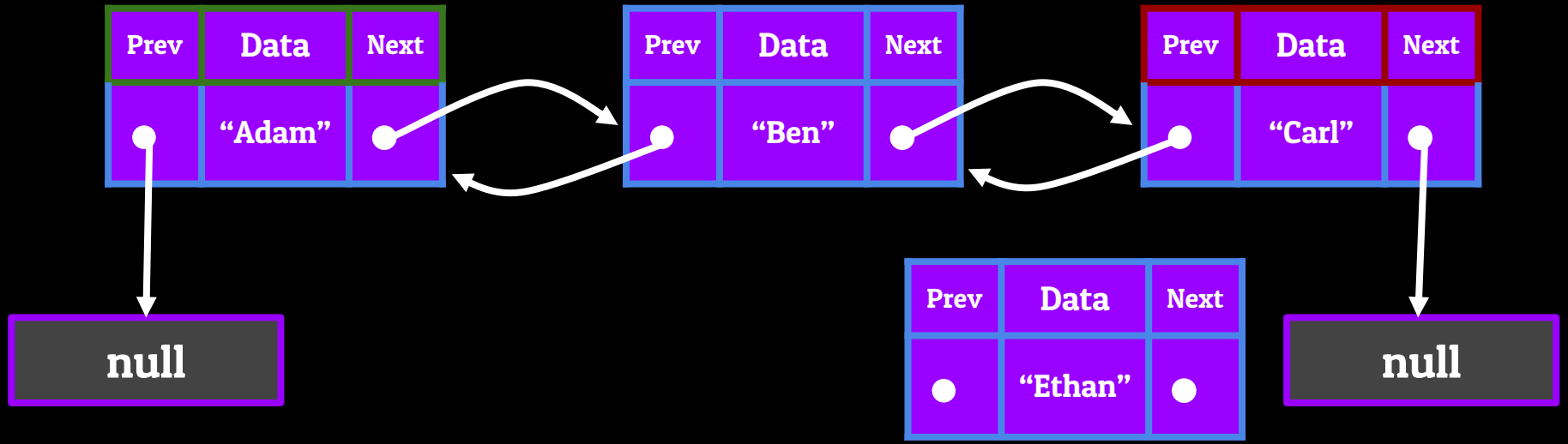
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



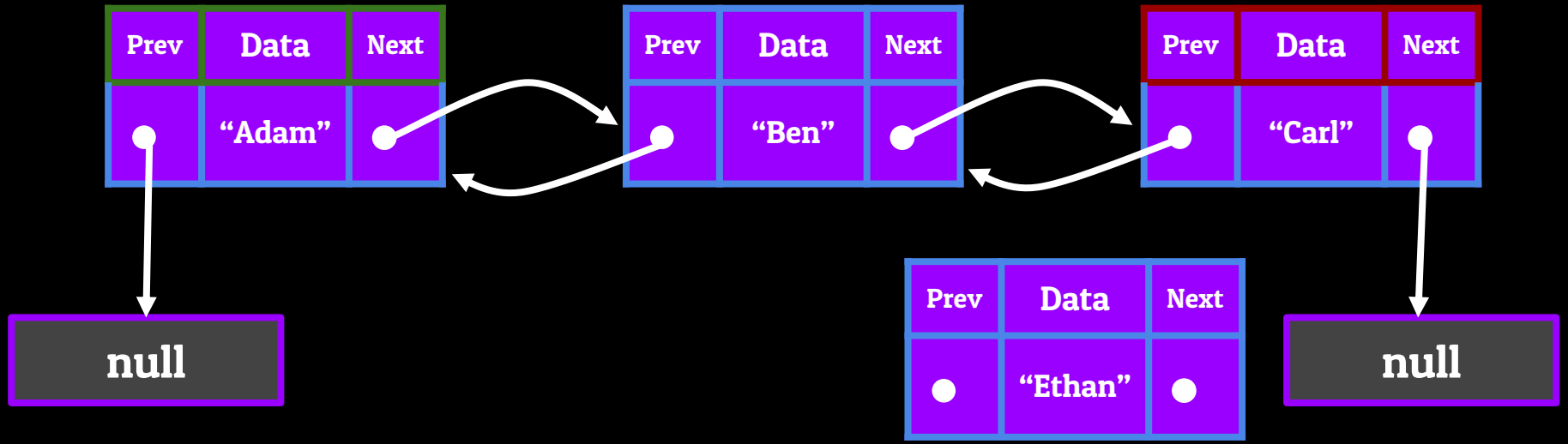
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



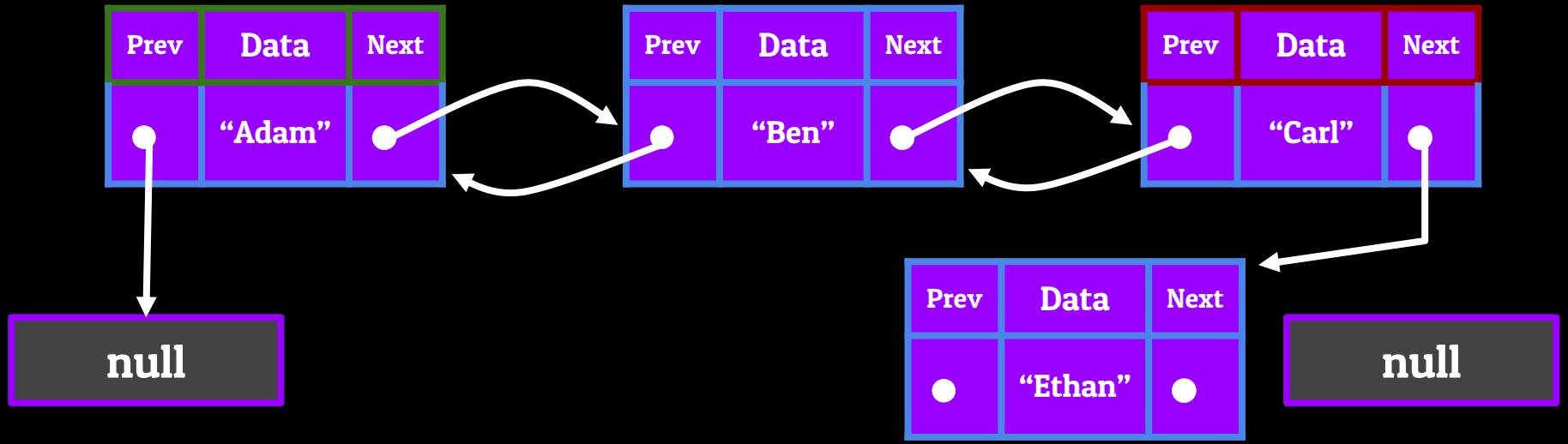
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



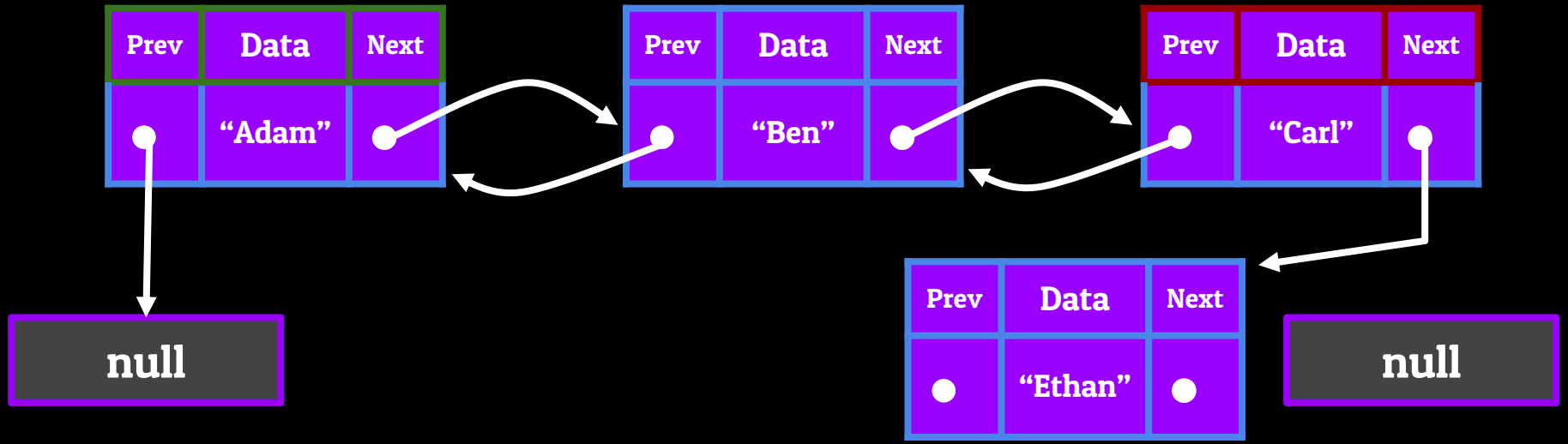
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



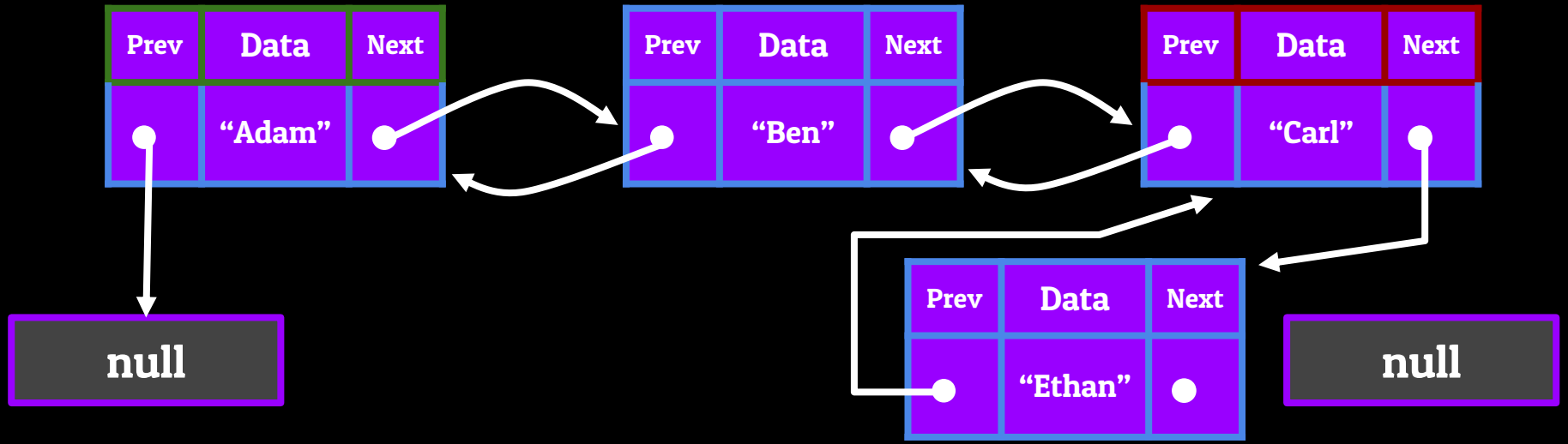
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



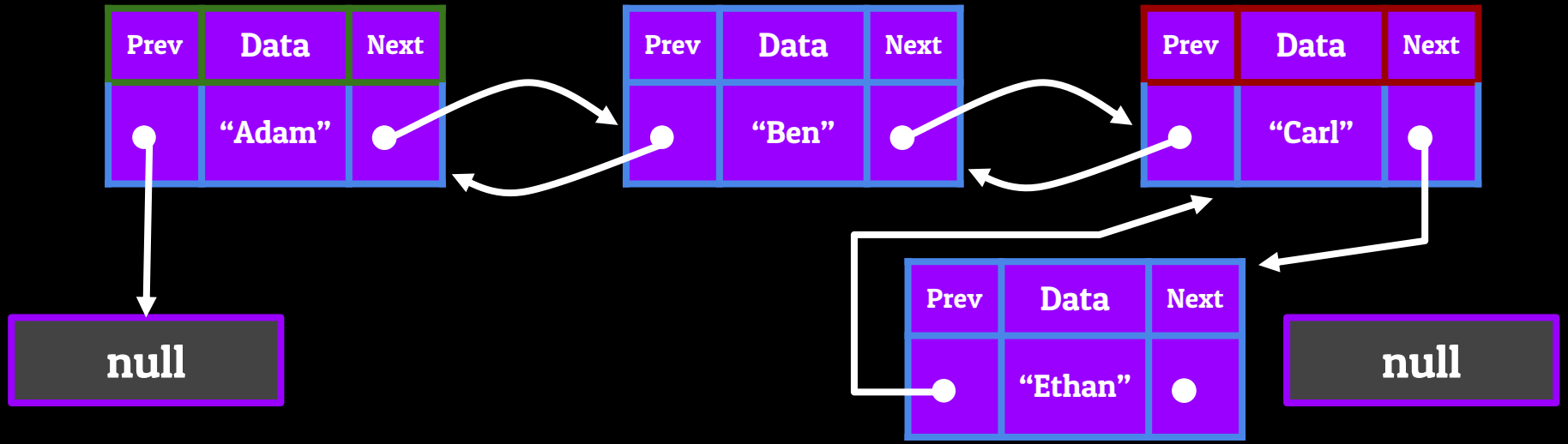
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



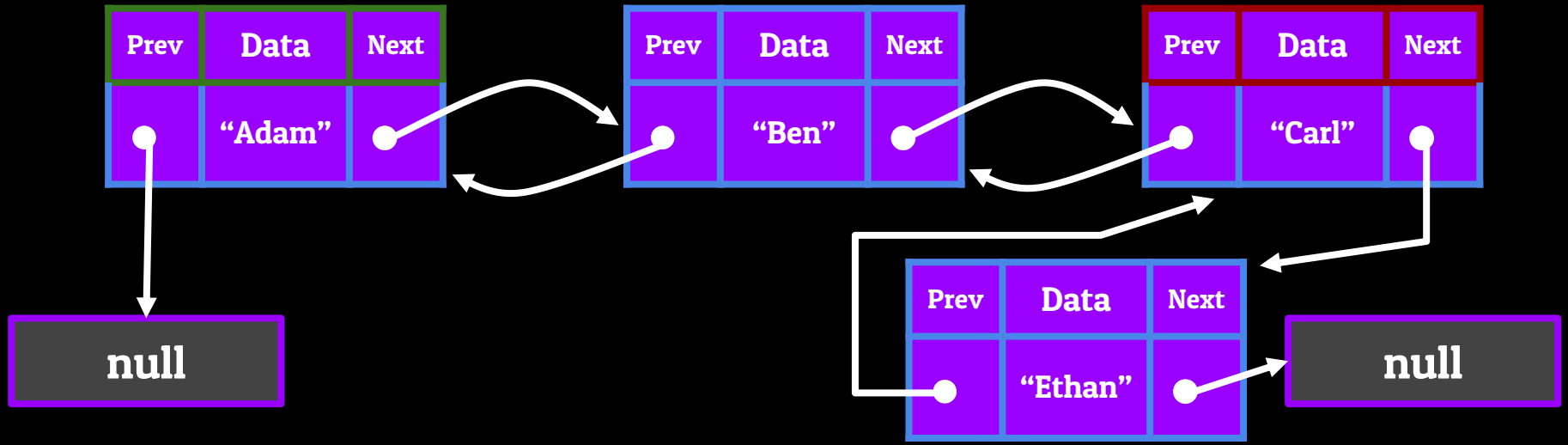
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value



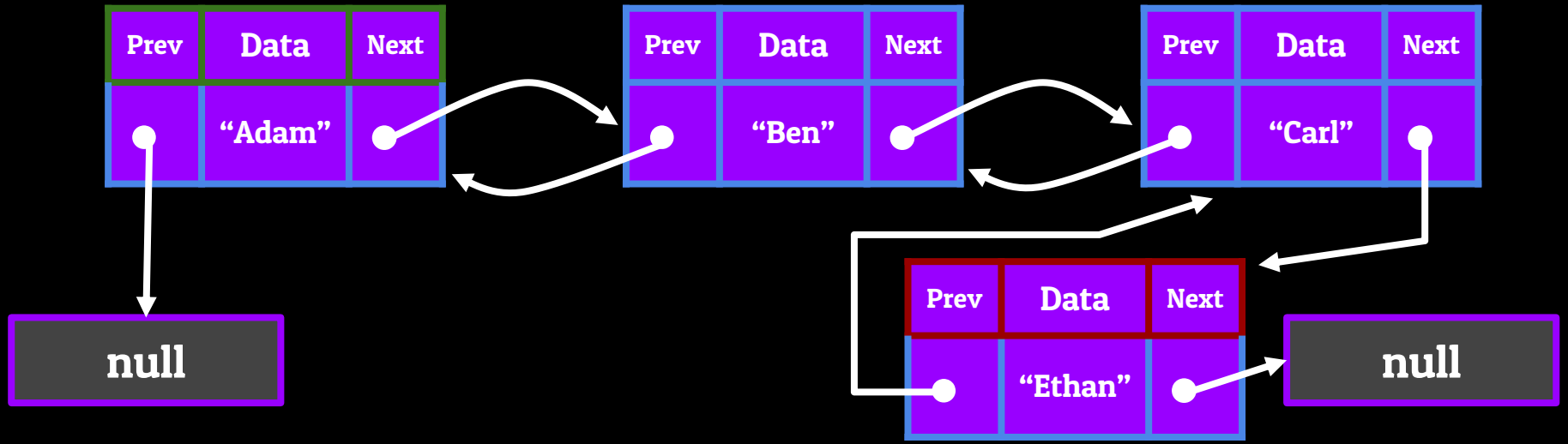
The Doubly-Linked List - Adding and Removing Information

Add to the Tail of a Doubly-LinkedList

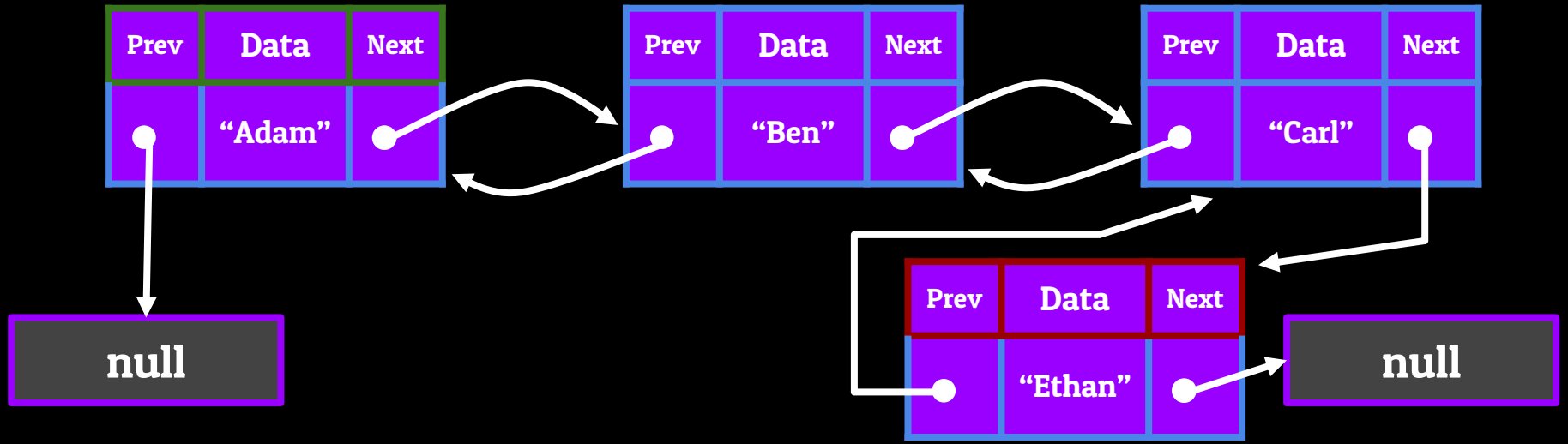
Set the next pointer of the current tail to point towards the new Node we want to become the tail

Set the previous of the new Node that we're adding to be pointing towards the current tail of the List

Make the new Node's next point towards a Null value

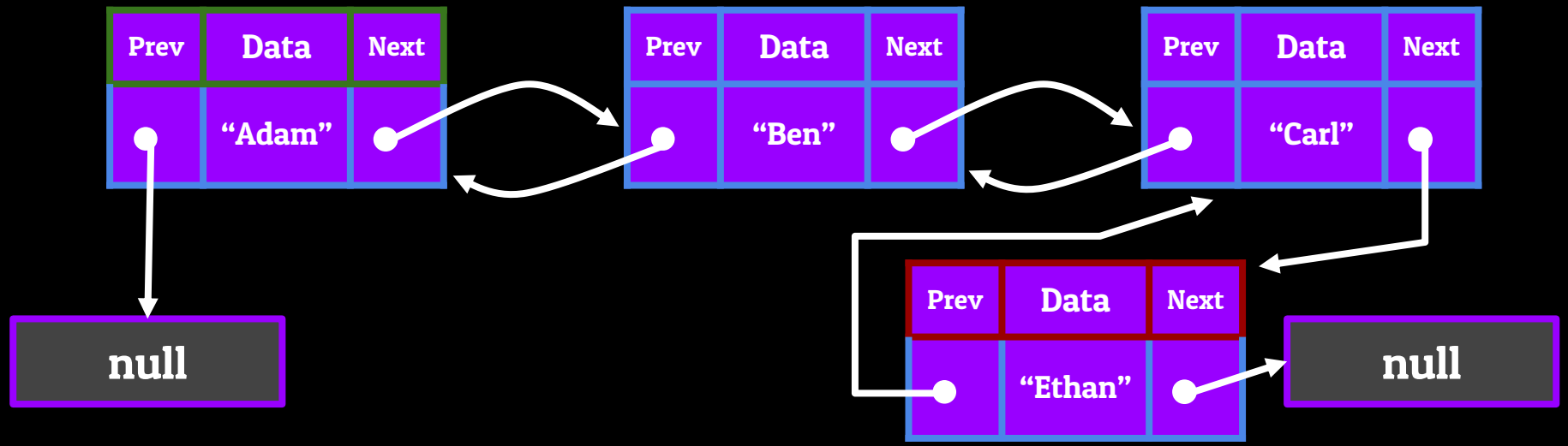


The Doubly-Linked List - Adding and Removing Information



The Doubly-Linked List - Adding and Removing Information

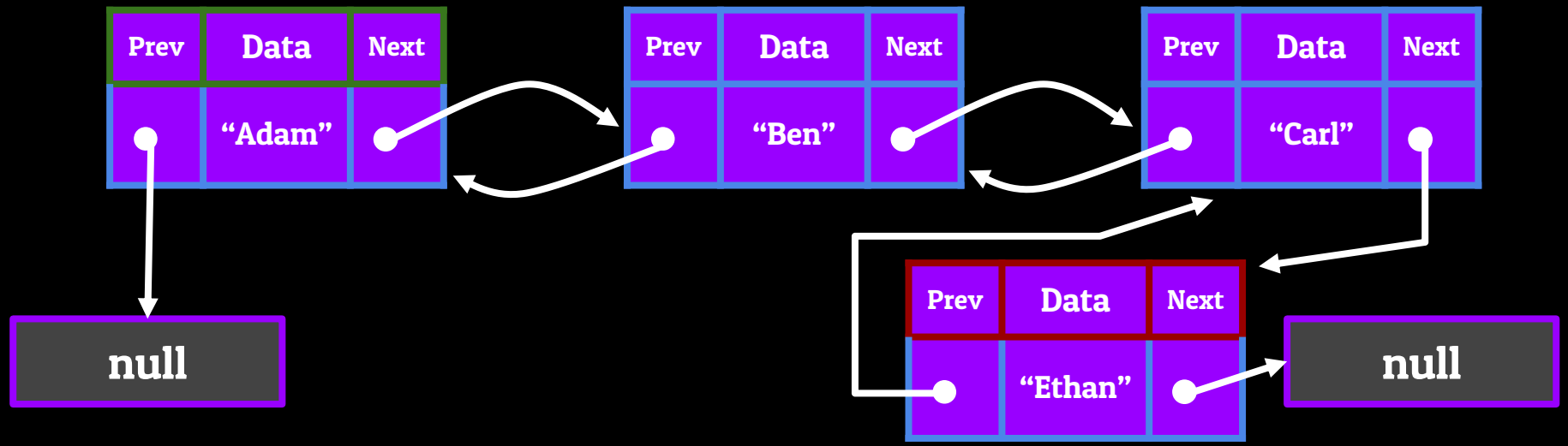
Remove from the Tail of a Doubly-LinkedList



The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

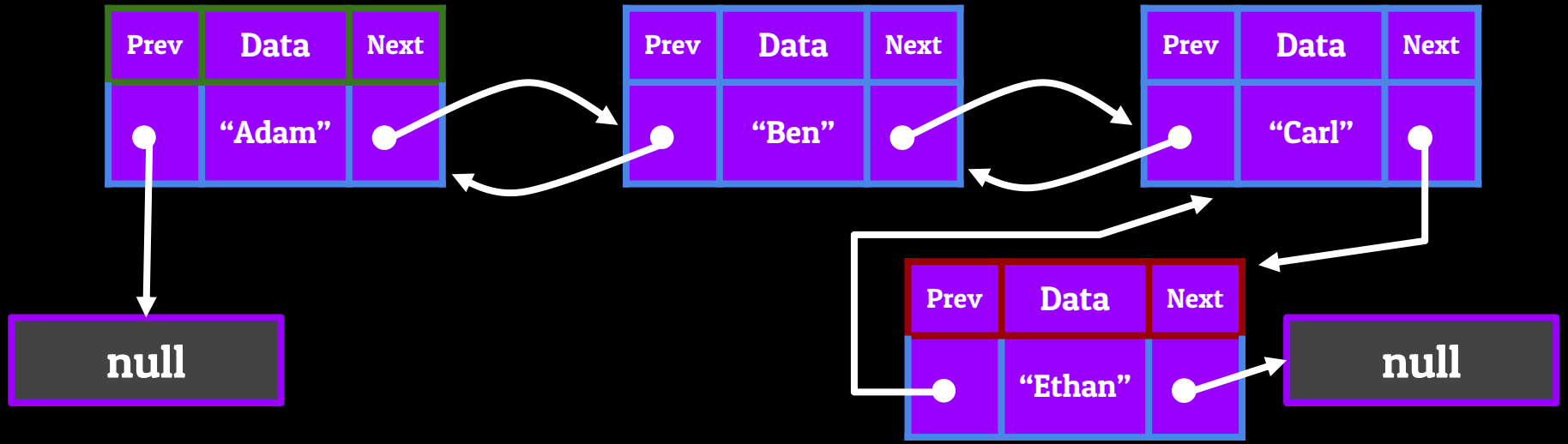


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

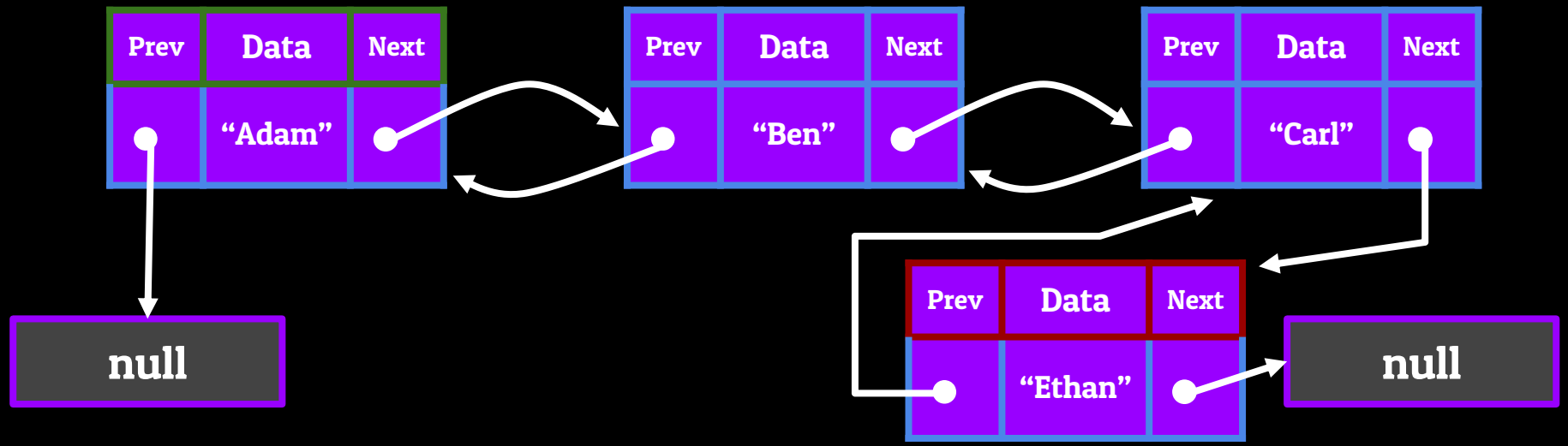


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

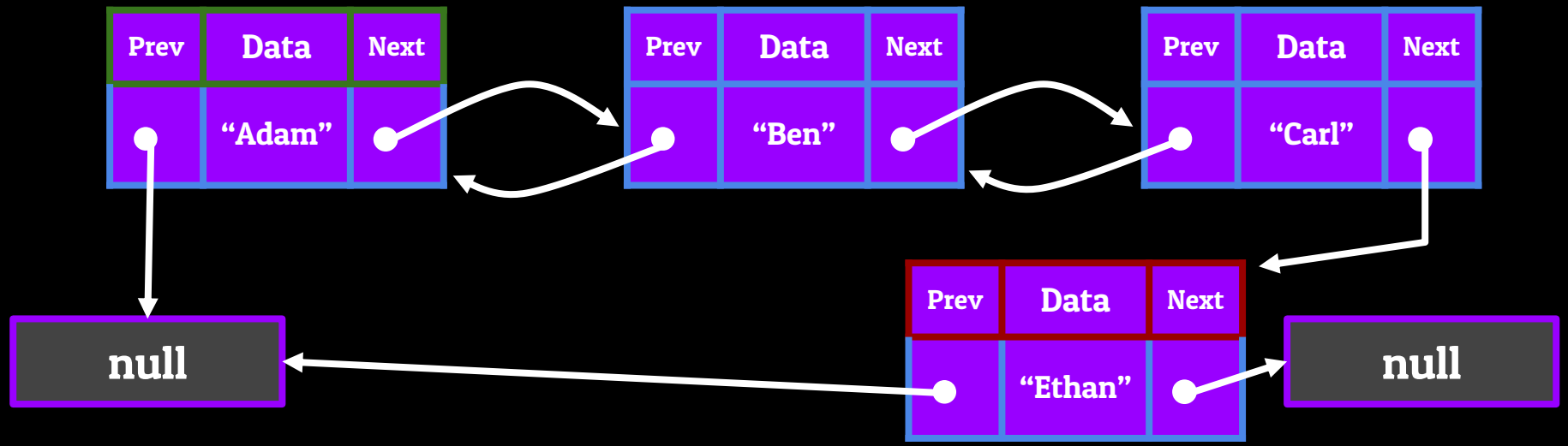


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

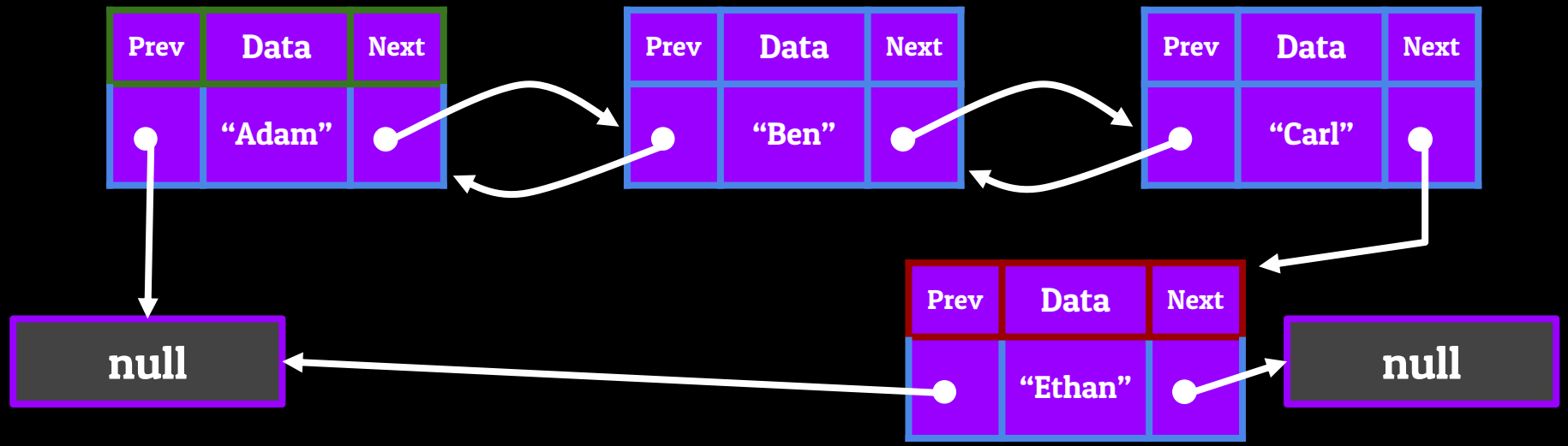


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

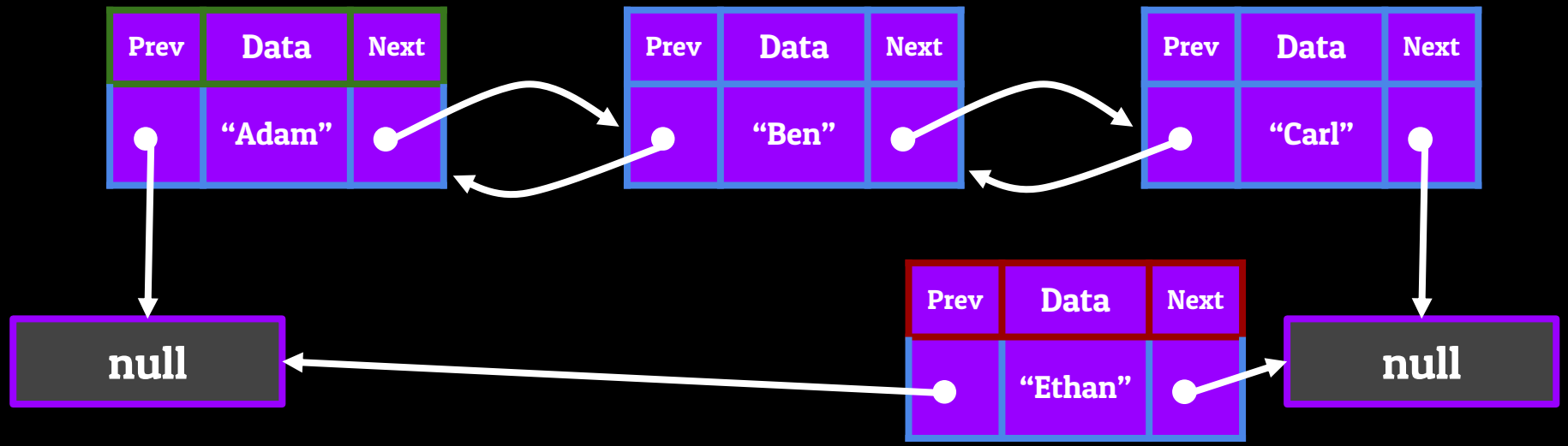


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

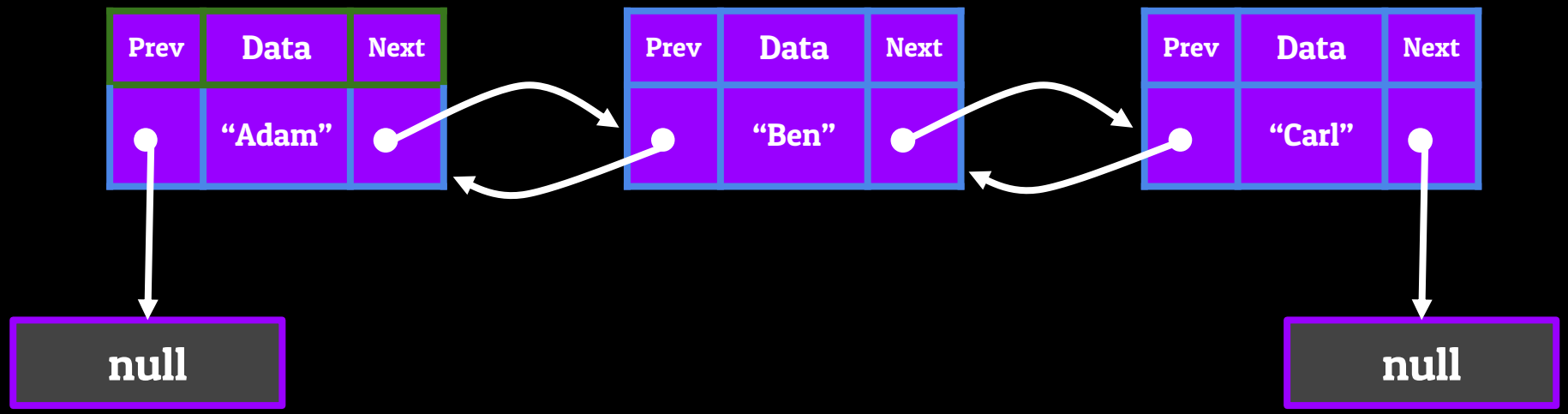


The Doubly-Linked List - Adding and Removing Information

Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

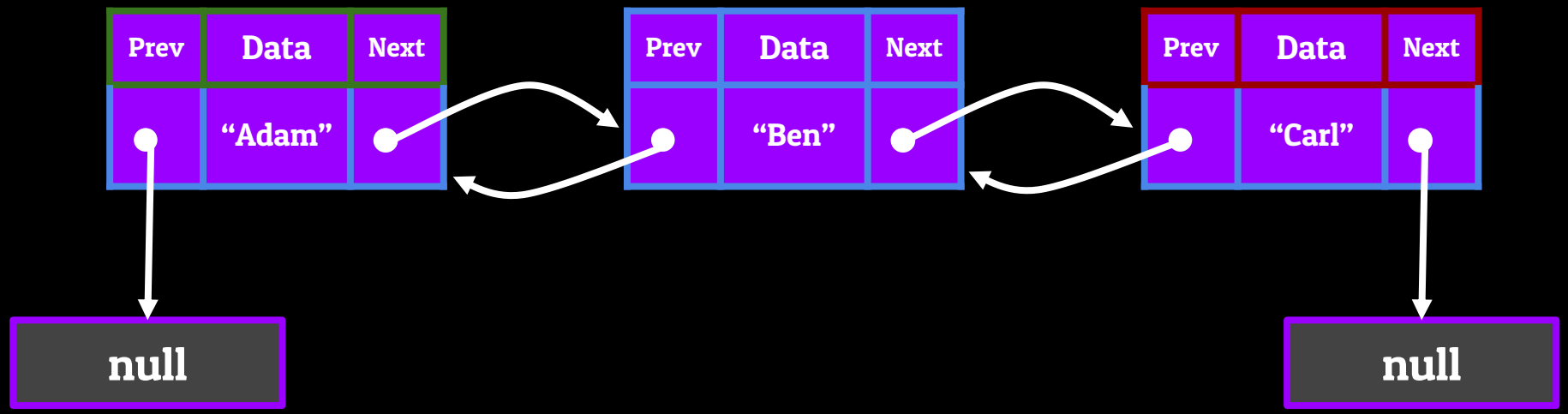


The Doubly-Linked List - Adding and Removing Information

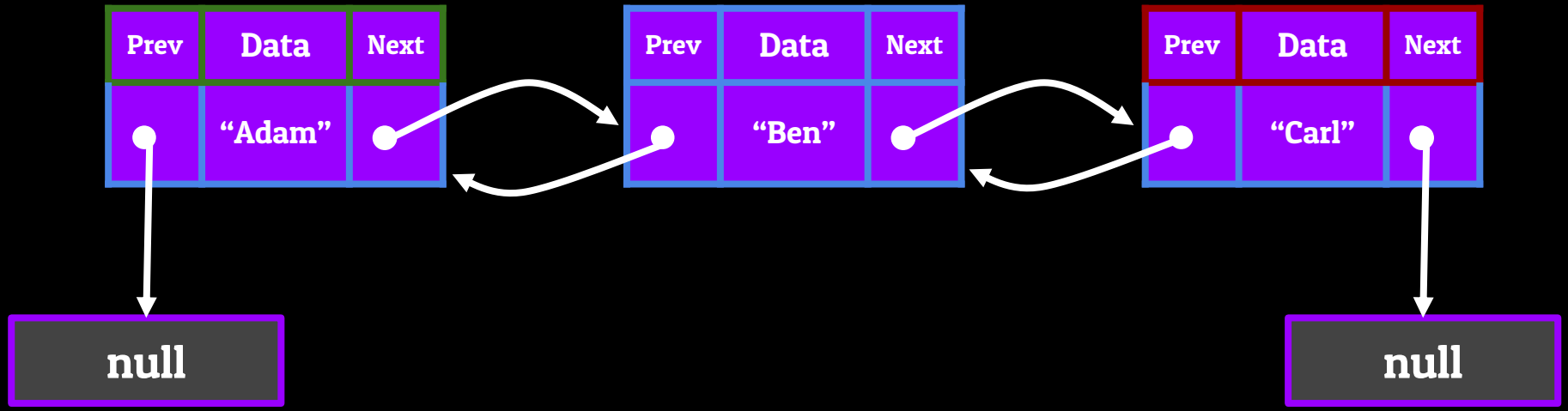
Remove from the Tail of a Doubly-LinkedList

Set the tail Node's previous to point towards null

Set the second to last Node's next to also point towards null

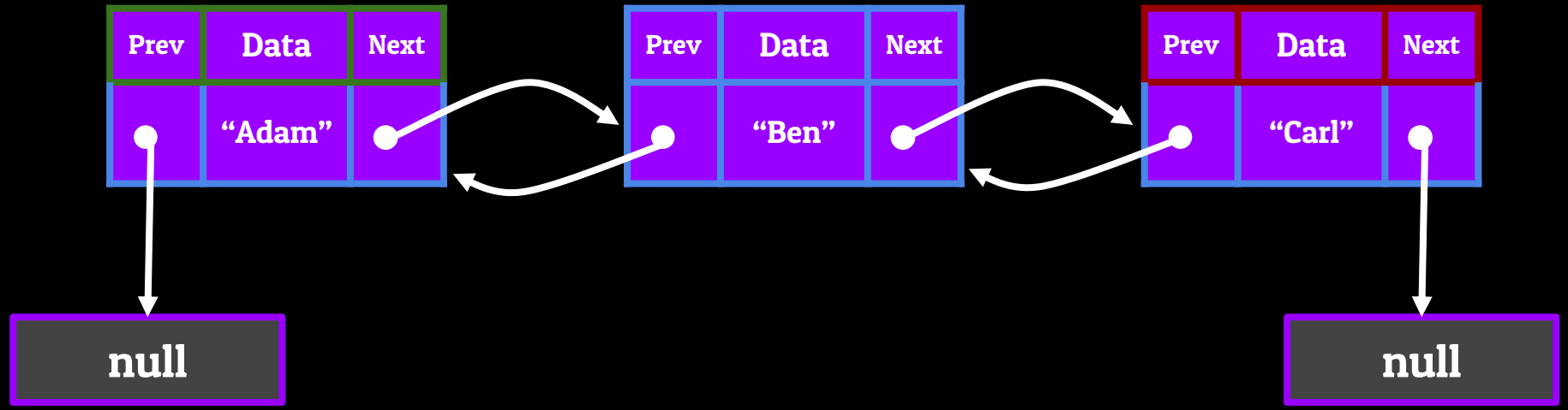


The Doubly-Linked List - Adding and Removing Information



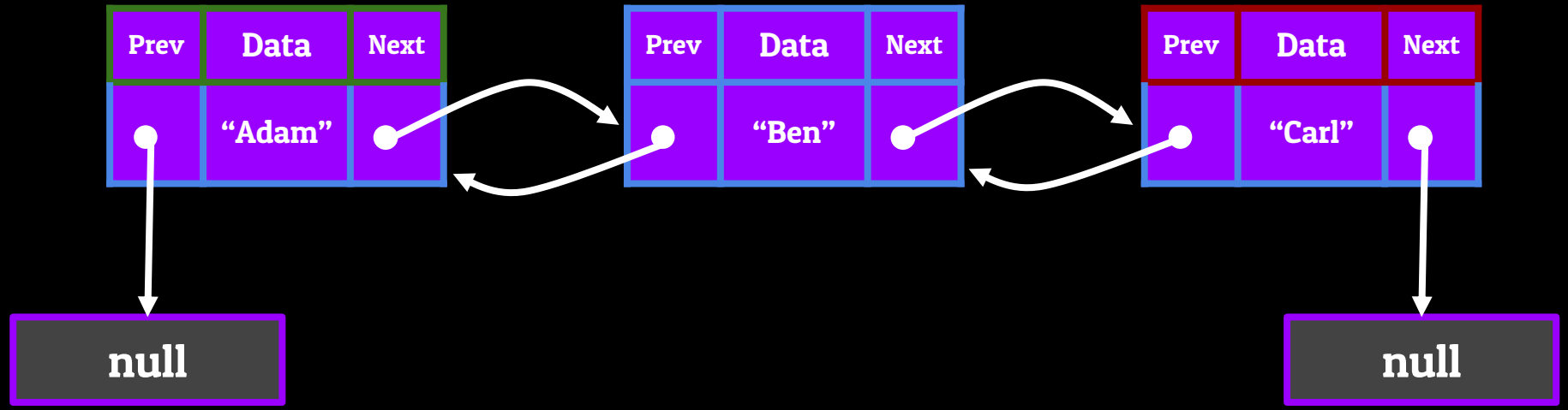
The Doubly-Linked List - Adding and Removing Information

- We only have to use this pseudocode to program these functions once, then we're able to use them an infinite amount of times



The Doubly-Linked List - Adding and Removing Information

- We only have to use this pseudocode to program these functions once, then we're able to use them an infinite amount of times

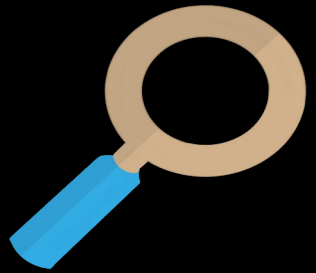


The Doubly-Linked List - Time Complexity Equations

The Doubly-Linked List - Time Complexity Equations

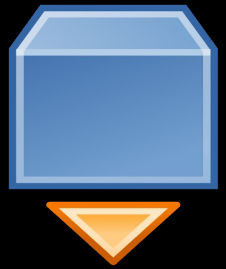
Accessing

O(n)



Searching

O(n)



Inserting

O(n) O(1)

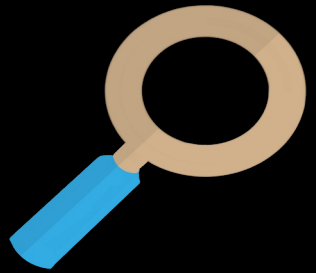
Deleting

O(n) O(1)

The Doubly-Linked List - Time Complexity Equations

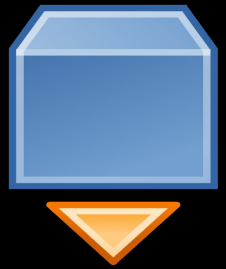
Accessing

O(n)



Searching

O(n)



Inserting

O(n) O(1)

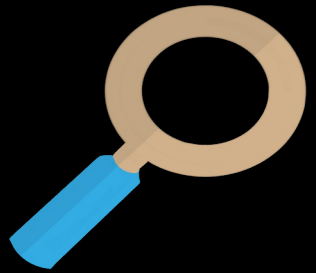
Deleting

O(n) O(1)

The Doubly-Linked List - Time Complexity Equations

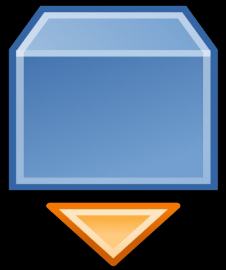
Accessing

O(n)



Searching

O(n)



Inserting

O(n) O(1)

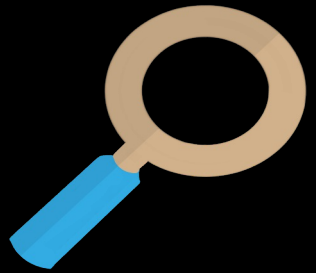
Deleting

O(n) O(1)

The Doubly-Linked List - Time Complexity Equations

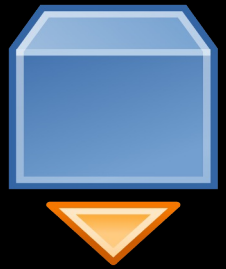
Accessing

O(n)



Searching

O(n)



Inserting

O(n) O(1)

Deleting

O(n) O(1)

The Doubly-Linked List - Uses of a Doubly-Linked List

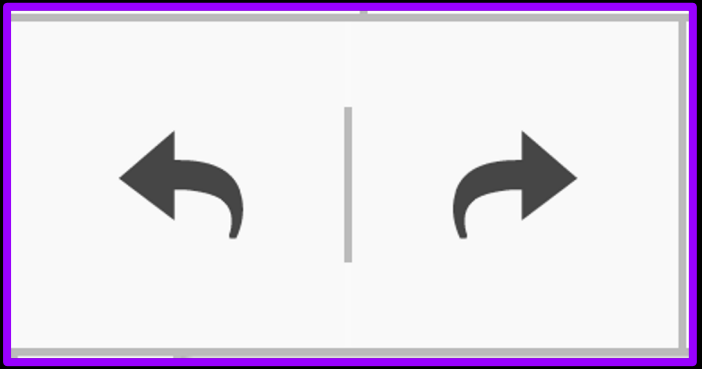
- The back and forth functionality of a Doubly-Linked List lends itself to be implemented in a lot of Stack-like functionality

The Doubly-Linked List - Uses of a Doubly-Linked List

- The back and forth functionality of a Doubly-Linked List lends itself to be implemented in a lot of Stack-like functionality

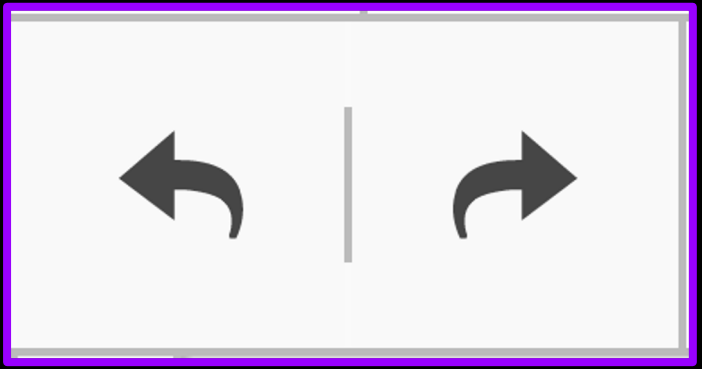
The Doubly-Linked List - Uses of a Doubly-Linked List

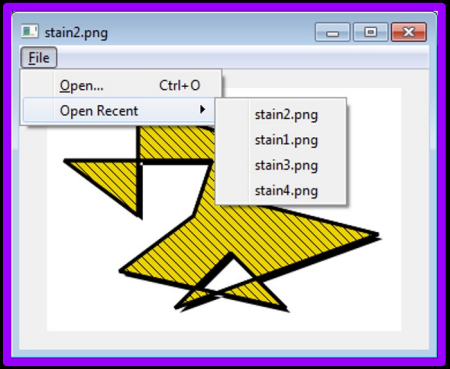
- The **back and forth functionality** of a Doubly-Linked List lends itself to be implemented in a lot of **Stack-like functionality**



The Doubly-Linked List - Uses of a Doubly-Linked List

- The **back and forth functionality** of a Doubly-Linked List lends itself to be implemented in a lot of **Stack-like functionality**

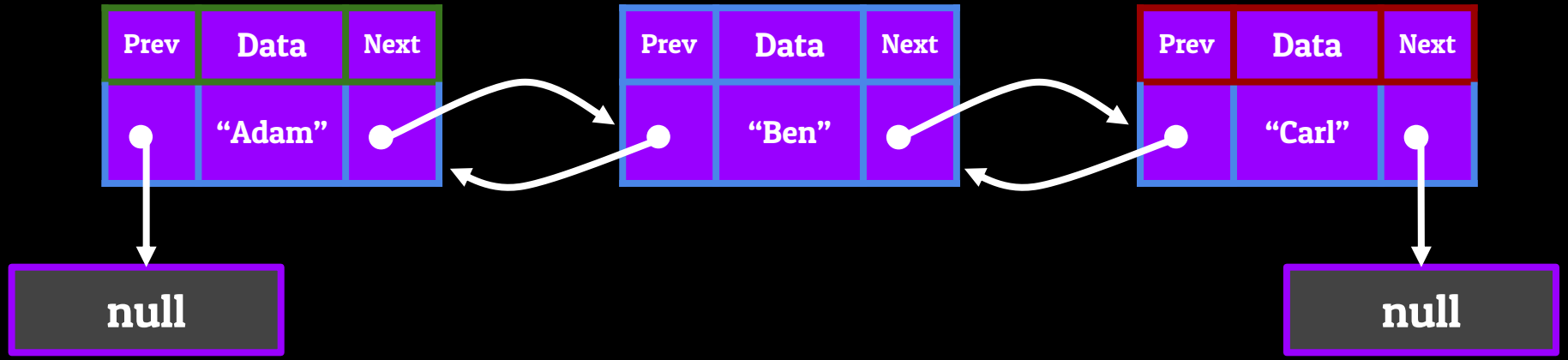




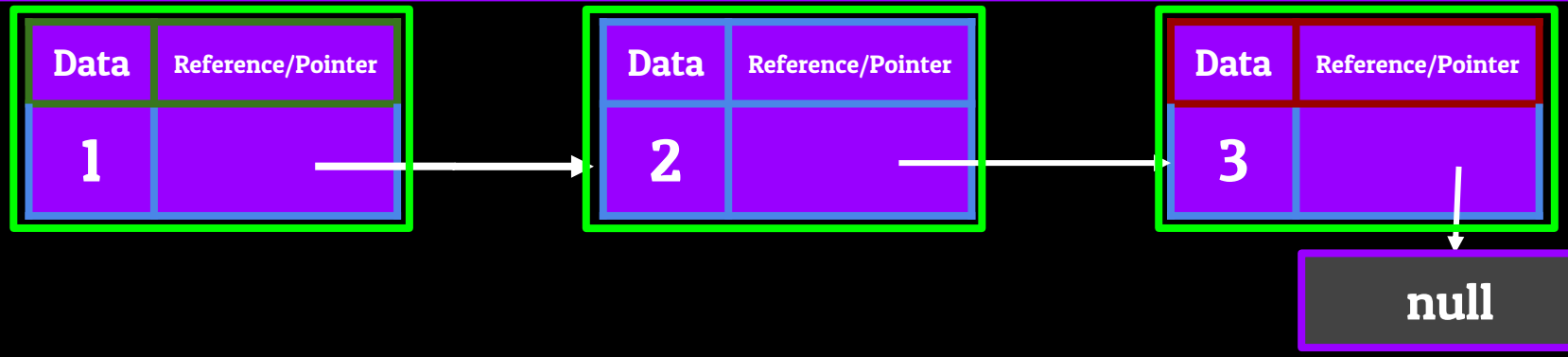
The Doubly-Linked List - Conclusion

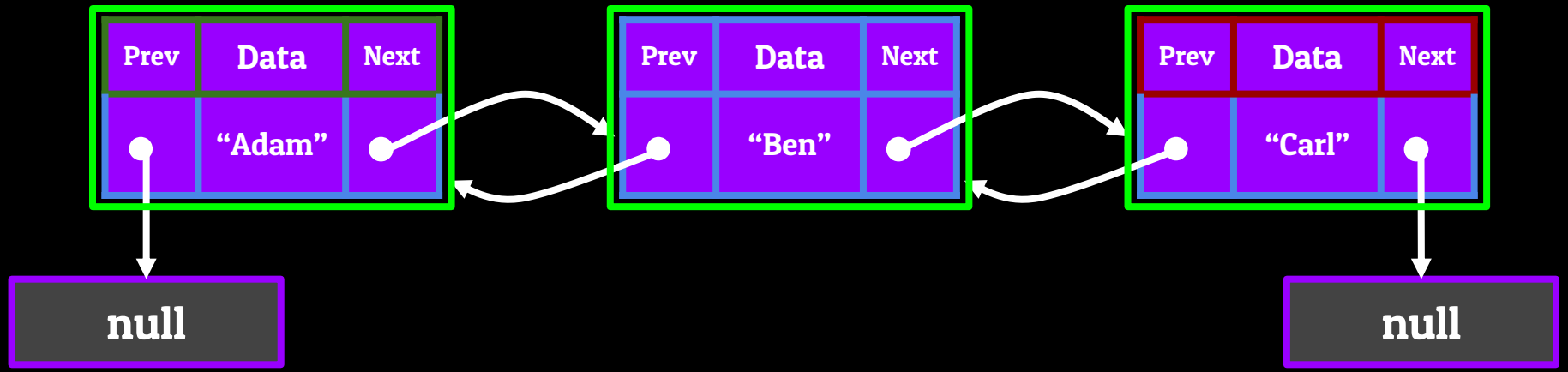
The Doubly-Linked List - Conclusion

The Doubly-Linked List - Conclusion

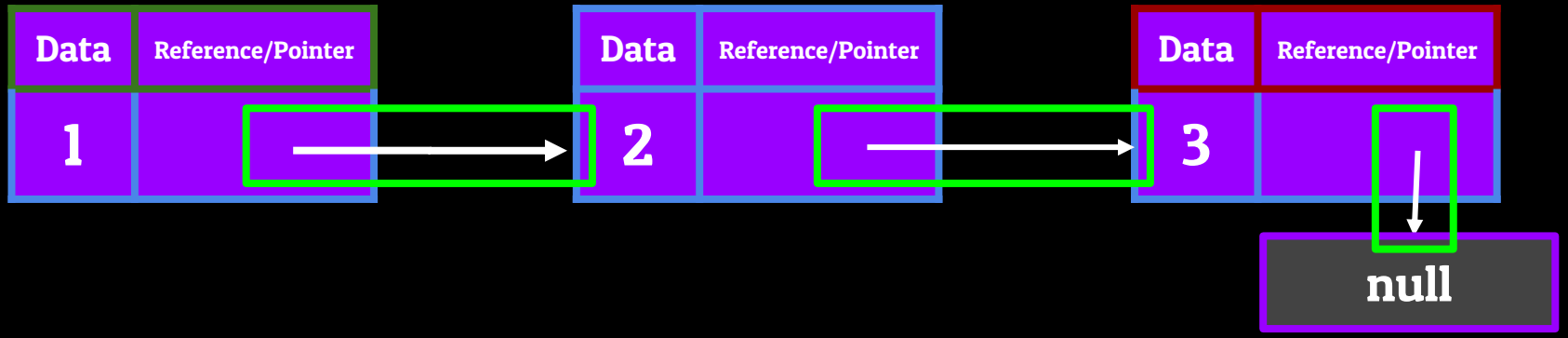


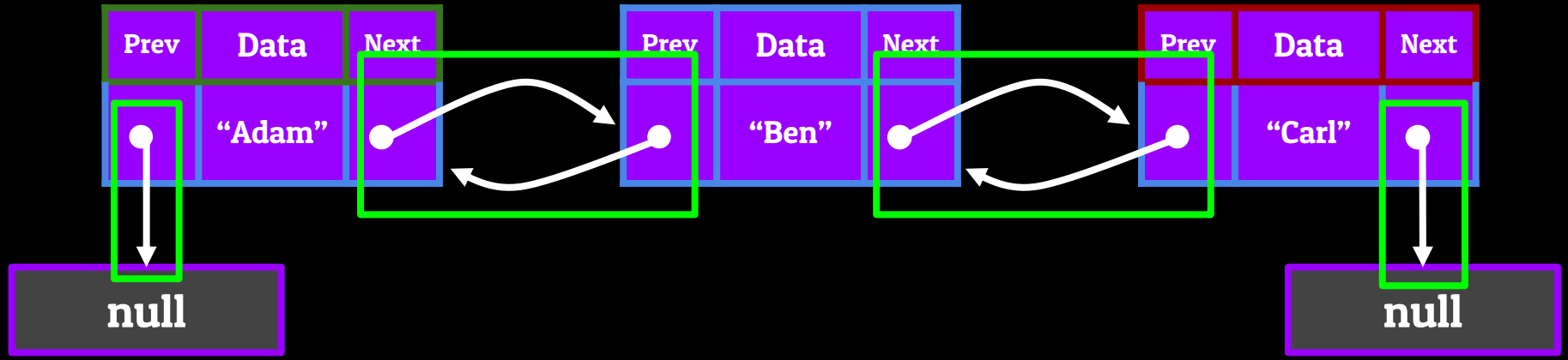
The Doubly-Linked List - Conclusion





The Doubly-Linked List - Conclusion





The Doubly-Linked List - Conclusion