The Stack

The stack is used to store data as code is executed. Every time a method is called a new area is added to the stack. This area is called a **frame**. Note that this is a **frame** for each time a method is called, not one frame for each method.

The frame for a method call stores the parameters passed in a method call and all the local variables used within that module. When the method finishes running the associated frame is deleted from the stack.

A Simple Java Application

```
public static void main(String[] args)
{
  int i=1;
}
```

- A frame is created on the stack
- The stack is filled from the top down (memory address)
- Within the main() frame all local variables are stored

simple java stack

```
</div>
```

Data is binary

```
public static void main(String[] args)
{
  int i=1;
}
```

- All data is stored in binary
- (shown here as one byte)
- (actual amount varies by chip)

simple java stack

```
</div>
```

Data stored smallest part first

```
public static void main(String[] args)
{
  int i=1;
}
```

- All data is stored in binary
- Is is stored as least significant "part" first
- (again, part size varies by chip)
- We'll typically just show the numbers in decimal
- We typically only need to deal with binary in Assembly

simple java stack

```
</div>
```