Code Breakdown of the Transformer

**1. Positional Encoding**

The **PositionalEncoding** class defines a positional encoding layer that adds positional information to the input sequences. It uses sine and cosine functions to create embeddings that represent the position of each token in the sequence.

**2. Multi-Head Attention**

The **MultiHeadAttention** class implements the multi-head attention mechanism. It consists of linear layers for query, key, and value projections, and performs scaled dot-product attention across multiple heads. The output is then linearly transformed.

**3. FeedForward**

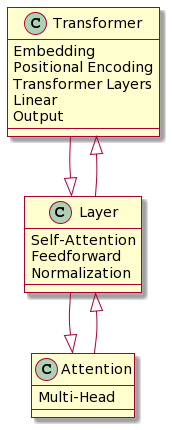
The **FeedForward** class defines a simple feedforward neural network used in each transformer layer. It consists of two linear layers with a ReLU activation in between.

**4. TransformerLayer**

The **TransformerLayer** class encapsulates a single layer of the Transformer model. It incorporates self-attention, feedforward, and layer normalization.

**5. Transformer**

The **Transformer** class brings together the aforementioned components to create a complete Transformer model. It includes an embedding layer, positional encoding, multiple transformer layers, and a linear output layer.



**Appendix on Training and Testing**

**Training Procedure**

The training loop involves iterating through batches of sequences from the dataset. For each batch:

1. The input sequence is split into source (**src**) and target (**tgt**) sequences.
2. A mask is created to ignore padding tokens in the source sequence.
3. The model is set to training mode, and the optimizer gradients are zeroed.
4. The model predicts the next tokens in the target sequence.
5. The CrossEntropyLoss is computed between the predicted and target sequences.
6. Backpropagation is performed, and the optimizer updates the model parameters.

**Testing Procedure**

The testing loop is similar but without backpropagation:

1. The model is set to evaluation mode.
2. The model predicts the next tokens in the target sequence.
3. The predictions are compared with the actual target sequence.
4. Accuracy is calculated as the ratio of correctly predicted tokens to the total tokens.

**Hyperparameters**

* **vocab\_size**: The size of the vocabulary.
* **d\_model**: Dimensionality of the model.
* **nhead**: Number of heads in the multi-head attention.
* **num\_layers**: Number of transformer layers.
* **d\_ff**: Dimensionality of the feedforward layer.