🏠 LangChain Expression Language How to

RunnablePassthrough: Passing data through

Passing data through

RunnablePassthrough allows to pass inputs unchanged or with the addition of extra keys. This typically is used in conjuction with RunnableParallel to assign data to a new key in the map.

RunnablePassthrough() called on it’s own, will simply take the input and pass it through.

RunnablePassthrough called with assign (`RunnablePassthrough.assign(...)`) will take the input, and will add the extra arguments passed to the assign function.

See the example below:

```
%pip install --upgrade --quiet  langchain
langchain-openai
```

```
from langchain_core.runnables import
RunnableParallel, RunnablePassthrough

runnable = RunnableParallel(
    passed=RunnablePassthrough(),
    extra=RunnablePassthrough.assign(mult=lambda x: x["num"] * 3),
    modified=lambda x: x["num"] + 1,
)

runnable.invoke({"num": 1})
```

```
{'passed': {'num': 1}, 'extra': {'num': 1, 'mult': 3}, 'modified': 2}
```

As seen above, `passed` key was called with `RunnablePassthrough()` and so it simply passed on `{'num': 1}`.

In the second line, we used `RunnablePastshrough.assign` with a lambda that multiplies the numerical value by 3. In this cased, `extra` was set with `{'num': 1, 'mult': 3}` which is the original value with the `mult` key added.

Finally, we also set a third key in the map with `modified` which uses a lambda to set a single value adding 1 to the num, which resulted in `modified` key with the value of `2`.

Retrieval Example

In the example below, we see a use case where we use RunnablePassthrough along with RunnableMap.

```
from langchain_community.vectorstores import FAISS
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.runnables import RunnablePassthrough
from langchain_openai import ChatOpenAI, OpenAIEmbeddings

vectorstore = FAISS.from_texts(
    ["harrison worked at kensho"], embedding=OpenAIEmbeddings()
)
retriever = vectorstore.as_retriever()
template = """Answer the question based only on the following context:
{context}

Question: {question}
"""
prompt = ChatPromptTemplate.from_template(template)
```

```
model = ChatOpenAI()

retrieval_chain = (
    {"context": retriever, "question": RunnablePassthrough()}
    | prompt
    | model
    | StrOutputParser()
)

retrieval_chain.invoke("where did harrison work?")
```

```
'Harrison worked at Kensho.'
```

Here the input to prompt is expected to be a map with keys “context” and “question”. The user input is just the question. So we need to get the context using our retriever and passthrough the user input under the “question” key. In this case, the RunnablePassthrough allows us to pass on the user’s question to the prompt and model.