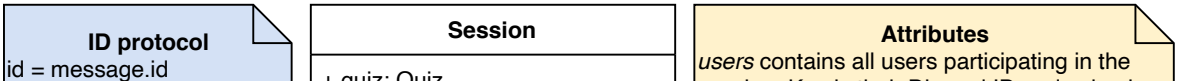
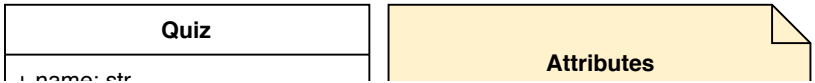
Quiz Administration & Editing

Abstract classes used by bot for administering/editing quizzes.



Quiz

Class to store questions and moderate them

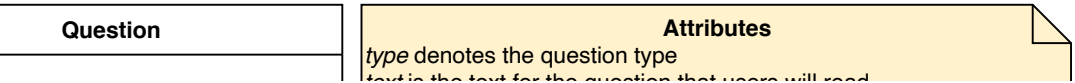


Qu

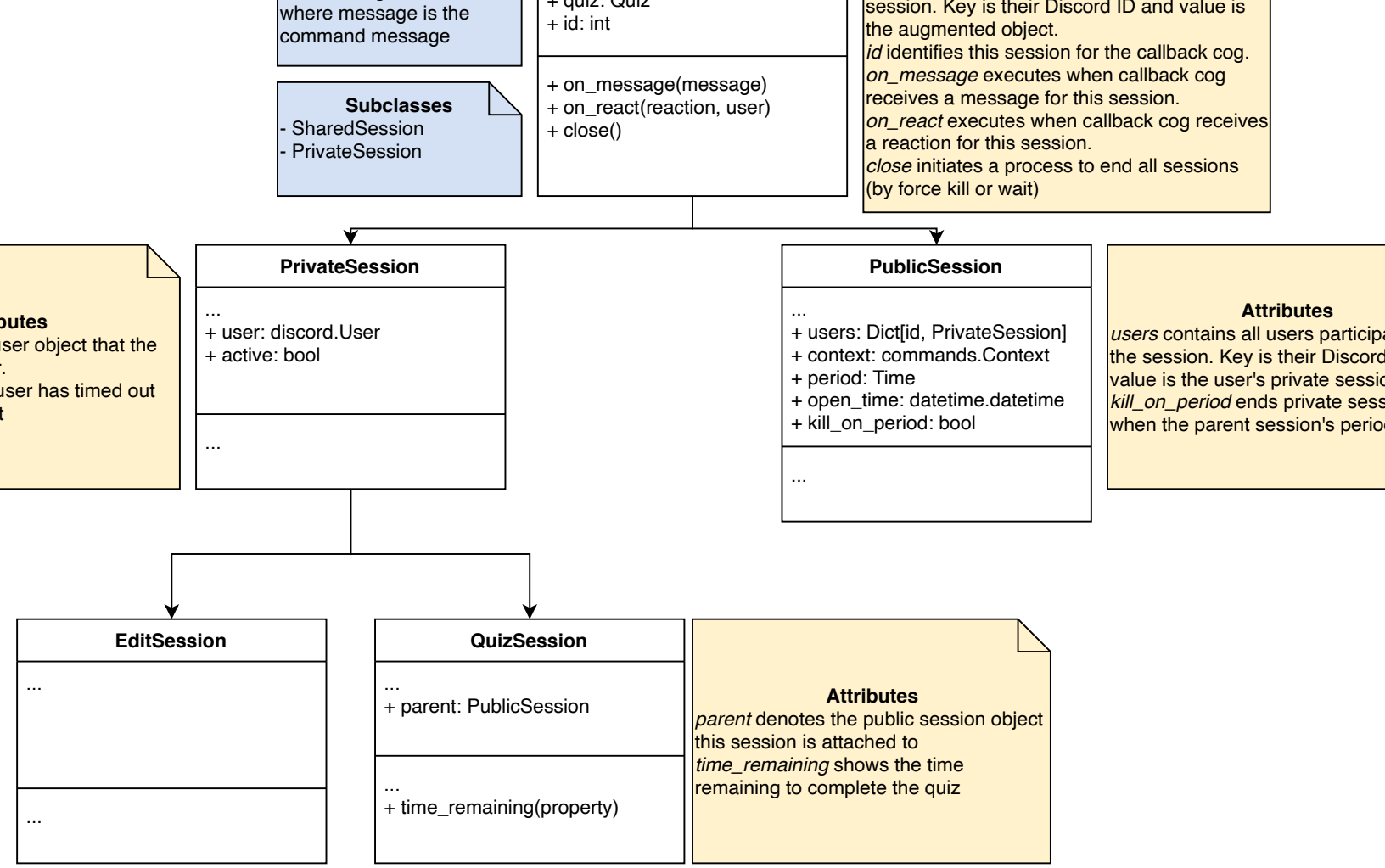
Class fra

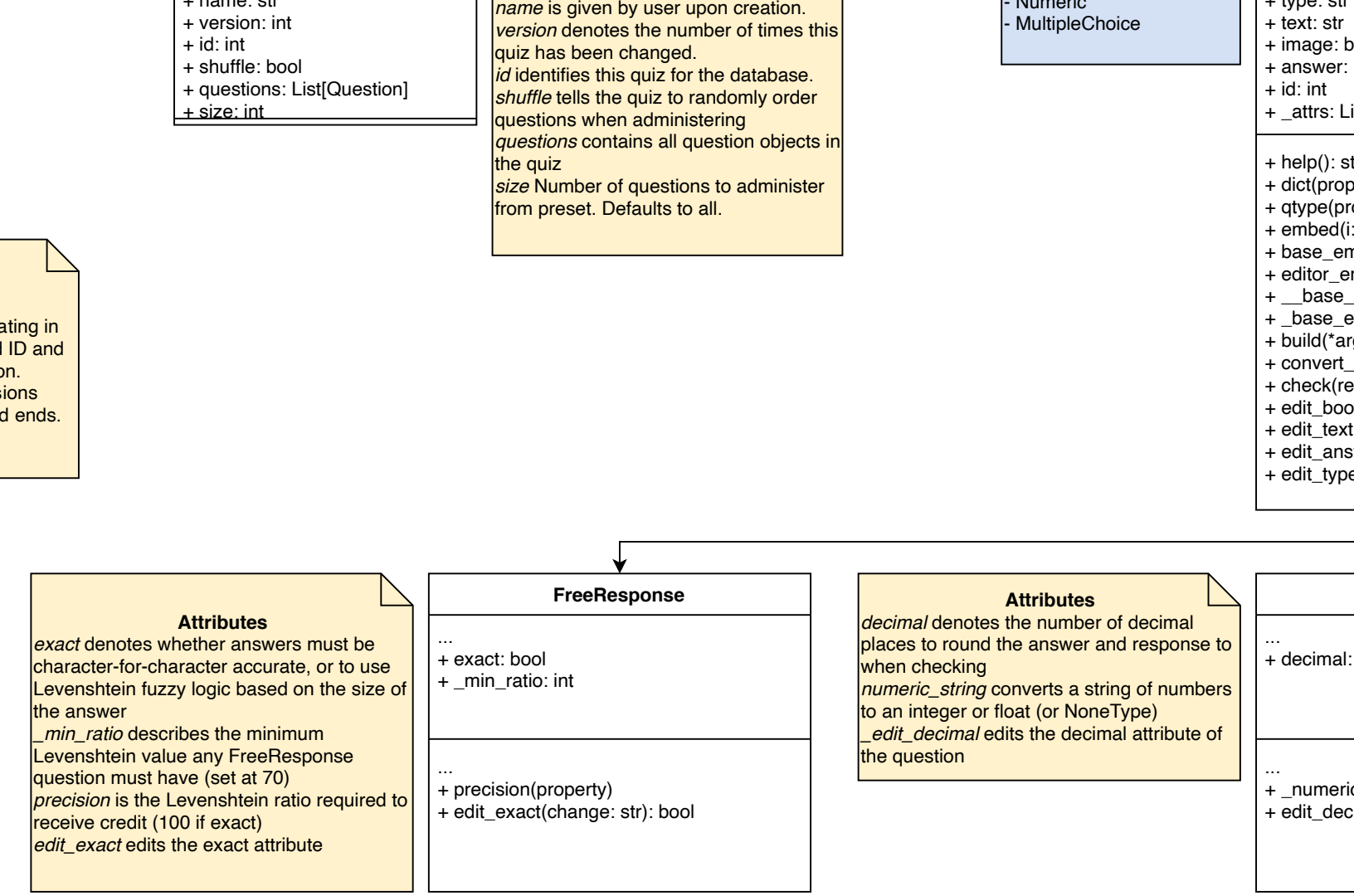
estion Types

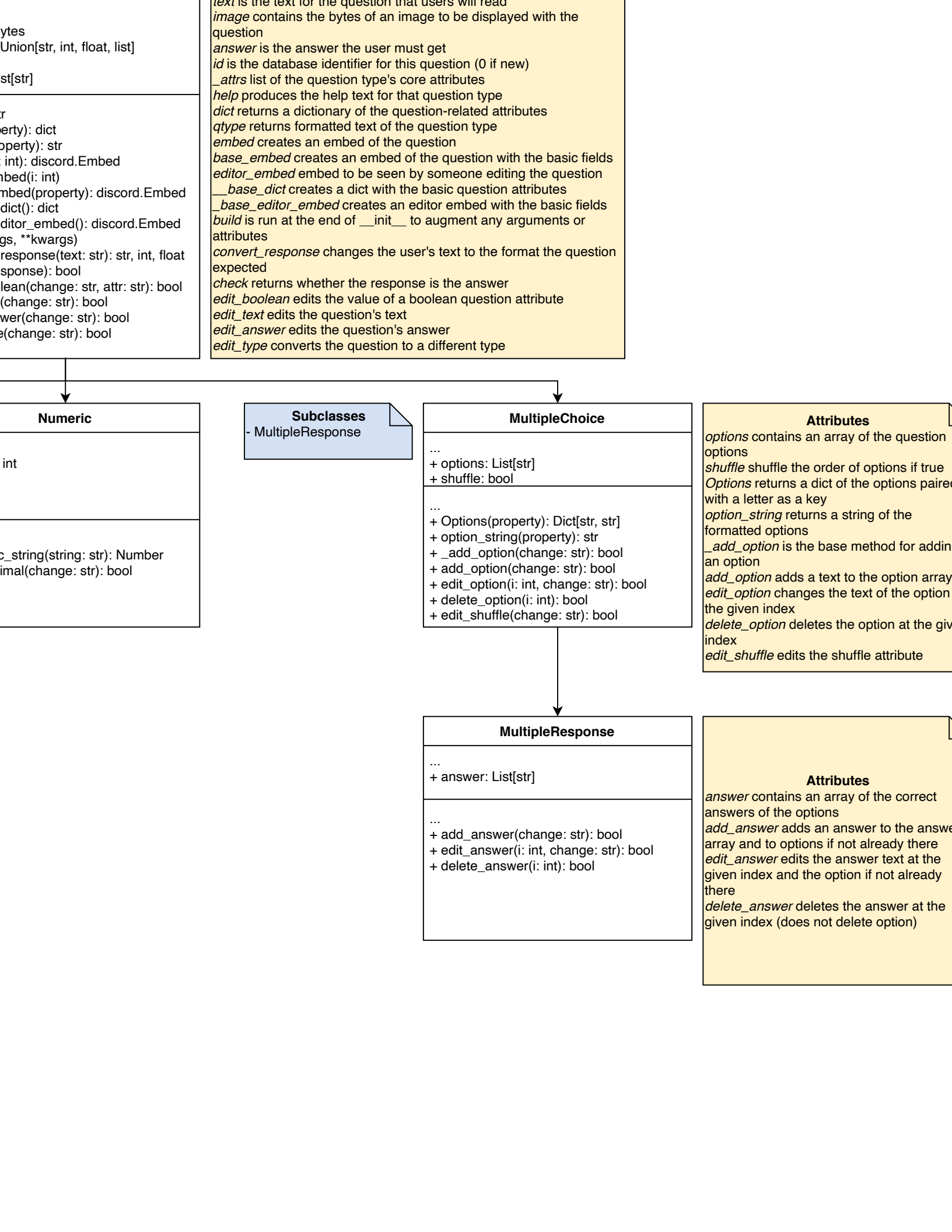
amework for offered questions

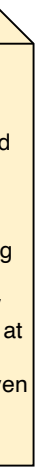


Attrib
user is the Discord u
private session is for
active is false if the u
during response wai


where message is the command message
+ quiz: Quiz
+ id: int
session. Key is their Discord ID and value is the augmented object.
id identifies this session for the callback cog.
on_message executes when callback cog receives a message for this session.
on_react executes when callback cog receives a reaction for this session.
close initiates a process to end all sessions (by force kill or wait)
Subclasses
- SharedSession
- PrivateSession
+ on_message(message)
+ on_react(reaction, user)
+ close()
PrivateSession
...
+ user: discord.User
+ active: bool
...
PublicSession
...
+ users: Dict[id, PrivateSession]
+ context: commands.Context
+ period: Time
+ open_time: datetime.datetime
+ kill_on_period: bool
...
Attributes
users contains all users participa
the session. Key is their Discord
value is the user's private sessio
kill_on_period ends private sess
when the parent session's perio
EditSession
...
...
QuizSession
...
+ parent: PublicSession
...
+ time_remaining(property)
Attributes
parent denotes the public session object this session is attached to
time_remaining shows the time remaining to complete the quiz


+ name: str
+ version: int
+ id: int
+ shuffle: bool
+ questions: List[Question]
+ size: int
name is given by user upon creation.
version denotes the number of times this quiz has been changed.
id identifies this quiz for the database.
shuffle tells the quiz to randomly order questions when administering
questions contains all question objects in the quiz
size Number of questions to administer from preset. Defaults to all.
- Numeric
- MultipleChoice
+ type: str
+ text: str
+ image: b
+ answer:
+ id: int
+ _attrs: Li
+ help(): st
+ dict(prop
+ qtype(pr
+ embed(i:
+ base_em
+ editor_e
+ __base_
+ _base_e
+ build(*ar
+ convert_
+ check(re
+ edit_boo
+ edit_text
+ edit_ans
+ edit_type
ating in
l ID and
on.
sions
d ends.
Attributes
exact denotes whether answers must be character-for-character accurate, or to use Levenshtein fuzzy logic based on the size of the answer
_min_ratio describes the minimum Levenshtein value any FreeResponse question must have (set at 70)
precision is the Levenshtein ratio required to receive credit (100 if exact)
edit_exact edits the exact attribute
FreeResponse
...
+ exact: bool
+ _min_ratio: int
...
+ precision(property)
+ edit_exact(change: str): bool
Attributes
decimal denotes the number of decimal places to round the answer and response to when checking
numeric_string converts a string of numbers to an integer or float (or NoneType)
_edit_decimal edits the decimal attribute of the question
...
+ decimal:
...
+ _numeric
+ edit_dec


ytes
Union[str, int, float, list]
st[str]
r
perty): dict
operty): str
int): discord.Embed
nbed(i: int)
mbed(property): discord.Embed
dict(): dict
ditor_embed(): discord.Embed
gs, **kwargs)
response(text: str): str, int, float
sponse): bool
lean(change: str, attr: str): bool
(change: str): bool
wer(change: str): bool
e(change: str): bool
text is the text for the question that users will read
image contains the bytes of an image to be displayed with the question
answer is the answer the user must get
id is the database identifier for this question (0 if new)
_attrs list of the question type's core attributes
help produces the help text for that question type
dict returns a dictionary of the question-related attributes
qtype returns formatted text of the question type
embed creates an embed of the question
base_embed creates an embed of the question with the basic fields
editor_embed embed to be seen by someone editing the question
__base_dict creates a dict with the basic question attributes
_base_editor_embed creates an editor embed with the basic fields
build is run at the end of __init__ to augment any arguments or attributes
convert_response changes the user's text to the format the question expected
check returns whether the response is the answer
edit_boolean edits the value of a boolean question attribute
edit_text edits the question's text
edit_answer edits the question's answer
edit_type converts the question to a different type
Numeric
int
c_string(string: str): Number
imal(change: str): bool
Subclasses
- MultipleResponse
MultipleChoice
...
+ options: List[str]
+ shuffle: bool
...
+ Options(property): Dict[str, str]
+ option_string(property): str
+ _add_option(change: str): bool
+ add_option(change: str): bool
+ edit_option(i: int, change: str): bool
+ delete_option(i: int): bool
+ edit_shuffle(change: str): bool
Attributes
options contains an array of the question options
shuffle shuffle the order of options if true
Options returns a dict of the options paire with a letter as a key
option_string returns a string of the formatted options
_add_option is the base method for addin an option
add_option adds a text to the option array
edit_option changes the text of the option the given index
delete_option deletes the option at the giv index
edit_shuffle edits the shuffle attribute
MultipleResponse
...
+ answer: List[str]
...
+ add_answer(change: str): bool
+ edit_answer(i: int, change: str): bool
+ delete_answer(i: int): bool
Attributes
answer contains an array of the correct answers of the options
add_answer adds an answer to the answe array and to options if not already there
edit_answer edits the answer text at the given index and the option if not already there
delete_answer deletes the answer at the given index (does not delete option)


d
g
at
ren


er