
nba_py Documentation

Release 0.1a2

nba_py

Nov 06, 2017

Contents

1	nba_py package	3
2	nba_py.player module	5
3	nba_py.game module	25
4	nba_py.team module	27
5	nba_py.constants module	35
6	Indices and tables	45
	Python Module Index	47

Contents:

CHAPTER 1

nba_py package

class `nba_py.Scoreboard` (*month=11, day=6, year=2017, league_id='00', offset=0*)
A scoreboard for all games for a given day Displays current games plus info for a given day

Args:

month Specified month (1-12)
day Specified day (1-31)
year Specified year (YYYY)
league_id ID for the league to look in (Default is 00)
offset Day offset from which to operate

Attributes:

json Contains the full json dump to play around with
available()
east_conf_standings_by_day()
game_header()
last_meeting()
line_score()
series_standings()
west_conf_standings_by_day()

CHAPTER 2

nba_py.player module

class nba_py.player.**PlayerCareer** (*player_id*, *per_mode*='PerGame', *league_id*='00')

Contains stats based on several parameters such as career regular season totals, post season career totals, all star season careers totals, college season career totals, etc.

Args:

player_id Player ID to look up

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

league_id ID for the league to look in (Default is 00)

Attributes:

json Contains the full json dump to play around with

all_star_season_totals ()

career_all_star_season_totals ()

college_season_career_totals ()

college_season_totals ()

post_season_career_totals ()

post_season_rankings ()

post_season_totals ()

preseason_career_totals ()

preseason_season_totals ()

regular_season_career_totals ()

regular_season_rankings ()

regular_season_totals ()

```
class nba_py.player.PlayerClutchSplits (player_id, team_id=0, measure_type='Base',
                                       per_mode='PerGame', plus_minus='N',
                                       pace_adjust='N', rank='N', league_id='00',
                                       season='2017-18', season_type='Regular Season',
                                       po_round='0', outcome='', location='', month='0',
                                       season_segment='', date_from='', date_to='', oppo-
                                       nent_team_id='0', vs_conference='', vs_division='',
                                       game_segment='', period='0', shot_clock_range='',
                                       last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains a lot of methods for last n minutes with a deficit of x points

Args:

- player_id** ID of the player to look up
- team_id** ID of the team to look up
- measure_type** Specifies type of measure to use (Base, Advanced, etc.)
- per_mode** Mode to measure statistics (Totals, PerGame, Per36, etc.)
- plus_minus** Whether or not to consider plus minus (Y or N)
- pace_adjust** Whether or not to pace adjust stats (Y or N)
- rank** Whether or not to consider rank (Y or N)
- league_id** ID for the league to look in (Default is 00)
- season** Season given to look up
- season_type** Season type to consider (Regular / Playoffs)
- po_round** Playoff round
- outcome** Filter out by wins or losses
- location** Filter out by home or away
- month** Specify month to filter by
- season_segment** Filter by pre/post all star break
- date_from** Filter out games before a specific date
- date_to** Filter out games after a specific date
- opponent_team_id** Opponent team ID to look up
- vs_conference** Filter by conference
- vs_division** Filter by division
- game_segment** Filter by half / overtime
- period** Filter by quarter / specific overtime
- shot_clock_range** Filter statistics by range in shot clock
- last_n_games** Filter by number of games specified in N

Attributes:

- json** Contains the full json dump to play around with

last10sec_deficit_3point ()
 Results in last 5 minutes <= 5 points

last1min_deficit_5point ()
Results in last 5 minutes <= 5 points

last1min_plusminus_5point ()
Last 1 minutes +/- 5 points

last30sec_deficit_3point ()
Results in last 5 minutes <= 5 points

last30sec_plusminus_5point ()
Last 30 seconds +/- 3 points

last3min_deficit_5point ()
Results in last 5 minutes <= 5 points

last3min_plusminus_5point ()
Last 3 minutes +/- 5 points

last5min_deficit_5point ()
Results in last 5 minutes <= 5 points

last5min_plusminus_5point ()
Last 5 minutes +/- 5 points

```
class nba_py.player.PlayerDefenseTracking(player_id, team_id=0, measure_type='Base',
                                         per_mode='PerGame', plus_minus='N',
                                         pace_adjust='N', rank='N', league_id='00',
                                         season='2017-18', season_type='Regular
                                         Season', po_round='0', outcome='', lo-
                                         cation='', month='0', season_segment='',
                                         date_from='', date_to='', oppo-
                                         nent_team_id='0', vs_conference='',
                                         vs_division='', game_segment='', period='0',
                                         shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Tracking data for defense for a given player

Args:

- player_id** ID of the player to look up
- team_id** ID of the team to look up
- measure_type** Specifies type of measure to use (Base, Advanced, etc.)
- per_mode** Mode to measure statistics (Totals, PerGame, Per36, etc.)
- plus_minus** Whether or not to consider plus minus (Y or N)
- pace_adjust** Whether or not to pace adjust stats (Y or N)
- rank** Whether or not to consider rank (Y or N)
- league_id** ID for the league to look in (Default is 00)
- season** Season given to look up
- season_type** Season type to consider (Regular / Playoffs)
- po_round** Playoff round
- outcome** Filter out by wins or losses
- location** Filter out by home or away

month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

```
class nba_py.player.PlayerGameLogs (player_id, league_id='00', season='2017-18', sea-
                                     son_type='Regular Season')
```

Contains a full log of all the games for a player for a given season

Args:

player_id ID of the player to look up
league_id ID for the league to look in (Default is 00)
season Season given to look up
season_type Season type to consider (Regular / Playoffs)

Attributes:

json Contains the full json dump to play around with

info()

```
class nba_py.player.PlayerGeneralSplits (player_id, team_id=0, measure_type='Base',
                                         per_mode='PerGame', plus_minus='N',
                                         pace_adjust='N', rank='N', league_id='00',
                                         season='2017-18', season_type='Regular Season',
                                         po_round='0', outcome='', location='', month='0',
                                         season_segment='', date_from='', date_to='', oppo-
                                         nent_team_id='0', vs_conference='', vs_division='',
                                         game_segment='', period='0', shot_clock_range='',
                                         last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains stats pertaining to location, wins and losses, pre/post all star break, starting position, and numbers of days rest

Args:

player_id ID of the player to look up
team_id ID of the team to look up
measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division

game_segment Filter by half / overtime

period Filter by quarter / specific overtime

shot_clock_range Filter statistics by range in shot clock

last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

days_rest ()

location ()

month ()

pre_post_all_star ()

starting_position ()

win_losses ()

```
class nba_py.player.PlayerInGameSplits (player_id, team_id=0, measure_type='Base',
                                         per_mode='PerGame', plus_minus='N',
                                         pace_adjust='N', rank='N', league_id='00',
                                         season='2017-18', season_type='Regular Season',
                                         po_round='0', outcome='', location='', month='0',
                                         season_segment='', date_from='', date_to='', oppo-
                                         nent_team_id='0', vs_conference='', vs_division='',
                                         game_segment='', period='0', shot_clock_range='',
                                         last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains player stats by half, by quarter, by score margin, and by actual margins.

Args:

player_id ID of the player to look up
team_id ID of the team to look up
measure_type Specifies type of measure to use (Base, Advanced, etc.)
per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)
plus_minus Whether or not to consider plus minus (Y or N)
pace_adjust Whether or not to pace adjust stats (Y or N)
rank Whether or not to consider rank (Y or N)
league_id ID for the league to look in (Default is 00)
season Season given to look up
season_type Season type to consider (Regular / Playoffs)
po_round Playoff round
outcome Filter out by wins or losses
location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with
by_actual_margin()
by_half()
by_period()
by_score_margin()

```
class nba_py.player.PlayerLastNGamesSplits (player_id, team_id=0, measure_type='Base',
                                             per_mode='PerGame', plus_minus='N',
                                             pace_adjust='N', rank='N', league_id='00',
                                             season='2017-18', season_type='Regular
                                             Season', po_round='0', outcome='', lo-
                                             cation='', month='0', season_segment='',
                                             date_from='', date_to='', oppo-
                                             nent_team_id='0', vs_conference='',
                                             vs_division='', game_segment='', period='0',
                                             shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains players stats per last 5, 10, 15, and 20 games, or specified number of games.

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division

game_segment Filter by half / overtime

period Filter by quarter / specific overtime

shot_clock_range Filter statistics by range in shot clock

last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

gamenumber ()

last10 ()

last15 ()

last20 ()

last5 ()

class nba_py.player.**PlayerList** (*league_id='00', season='2017-18', only_current=1*)

Contains a list of all players for a season, if specified, and will only contain current players if specified as well

Args:

league_id ID for the league to look in (Default is 00)

season Season given to look up

only_current Restrict lookup to only current players

Attributes:

json Contains the full json dump to play around with

info ()

exception nba_py.player.**PlayerNotFoundException**

Bases: exceptions.Exception

class nba_py.player.**PlayerOpponentSplits** (*player_id, team_id=0, measure_type='Base', per_mode='PerGame', plus_minus='N', pace_adjust='N', rank='N', league_id='00', season='2017-18', season_type='Regular Season', po_round='0', outcome='', location='', month='0', season_segment='', date_from='', date_to='', opponent_team_id='0', vs_conference='', vs_division='', game_segment='', period='0', shot_clock_range='', last_n_games='0')*)

Bases: nba_py.player._PlayerDashboard

Contains stats pertaining to player stats vs certain opponents by division, conference, and by specific team opponent

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

by_conference ()

by_division ()

by_opponent ()

```
class nba_py.player.PlayerPassTracking (player_id, team_id=0, measure_type='Base',
                                       per_mode='PerGame', plus_minus='N',
                                       pace_adjust='N', rank='N', league_id='00',
                                       season='2017-18', season_type='Regular Season',
                                       po_round='0', outcome='', location='', month='0',
                                       season_segment='', date_from='', date_to='', oppo-
                                       nent_team_id='0', vs_conference='', vs_division='',
                                       game_segment='', period='0', shot_clock_range='',
                                       last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Tracking data for passing for a given player

Args:

player_id ID of the player to look up
team_id ID of the team to look up
measure_type Specifies type of measure to use (Base, Advanced, etc.)
per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)
plus_minus Whether or not to consider plus minus (Y or N)
pace_adjust Whether or not to pace adjust stats (Y or N)
rank Whether or not to consider rank (Y or N)
league_id ID for the league to look in (Default is 00)
season Season given to look up
season_type Season type to consider (Regular / Playoffs)

po_round Playoff round
outcome Filter out by wins or losses
location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

passes_made ()

passes_received ()

```
class nba_py.player.PlayerPerformanceSplits (player_id, team_id=0, measure_type='Base',
                                             per_mode='PerGame', plus_minus='N',
                                             pace_adjust='N', rank='N', league_id='00',
                                             season='2017-18', season_type='Regular
                                             Season', po_round='0', outcome='', lo-
                                             cation='', month='0', season_segment='',
                                             date_from='', date_to='', oppo-
                                             nent_team_id='0', vs_conference='',
                                             vs_division='', game_segment='', period='0',
                                             shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Player stats by different performance metrics such as score differential, points scored, and points scored against

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)
po_round Playoff round
outcome Filter out by wins or losses
location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

points_against ()

points_scored ()

score_differential ()

class `nba_py.player.PlayerProfile` (*player_id*, *per_mode*='PerGame', *league_id*='00')

Bases: `nba_py.player.PlayerCareer`

Contains a more in depth version of player career stats with season highs, career highs, and when the player's next game is

Args:

player_id Player ID to look up

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

league_id ID for the league to look in (Default is 00)

Attributes:

json Contains the full json dump to play around with

career_highs ()

next_game ()

season_highs ()

```
class nba_py.player.PlayerReboundLogTracking(player_id, team_id=0, measure_type='Base',  
                                             per_mode='PerGame', plus_minus='N',  
                                             pace_adjust='N', rank='N', league_id='00',  
                                             season='2017-18', season_type='Regular  
                                             Season', po_round='0', outcome='', location='',  
                                             month='0', season_segment='',  
                                             date_from='', date_to='', oppo-  
                                             nent_team_id='0', vs_conference='',  
                                             vs_division='', game_segment='', period='0',  
                                             shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains a log for every rebound for a given season for a given player

Args:

- player_id** ID of the player to look up
- team_id** ID of the team to look up
- measure_type** Specifies type of measure to use (Base, Advanced, etc.)
- per_mode** Mode to measure statistics (Totals, PerGame, Per36, etc.)
- plus_minus** Whether or not to consider plus minus (Y or N)
- pace_adjust** Whether or not to pace adjust stats (Y or N)
- rank** Whether or not to consider rank (Y or N)
- league_id** ID for the league to look in (Default is 00)
- season** Season given to look up
- season_type** Season type to consider (Regular / Playoffs)
- po_round** Playoff round
- outcome** Filter out by wins or losses
- location** Filter out by home or away
- month** Specify month to filter by
- season_segment** Filter by pre/post all star break
- date_from** Filter out games before a specific date
- date_to** Filter out games after a specific date
- opponent_team_id** Opponent team ID to look up
- vs_conference** Filter by conference
- vs_division** Filter by division
- game_segment** Filter by half / overtime
- period** Filter by quarter / specific overtime
- shot_clock_range** Filter statistics by range in shot clock
- last_n_games** Filter by number of games specified in N

Attributes:

- json** Contains the full json dump to play around with

```
class nba_py.player.PlayerReboundTracking(player_id, team_id=0, measure_type='Base',
per_mode='PerGame', plus_minus='N',
pace_adjust='N', rank='N', league_id='00',
season='2017-18', season_type='Regular Season',
po_round='0', outcome='', location='',
month='0', season_segment='', date_from='',
date_to='', opponent_team_id='0', vs_conference='',
vs_division='', game_segment='', period='0',
shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Tracking data for rebounding for a given player

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division

game_segment Filter by half / overtime

period Filter by quarter / specific overtime

shot_clock_range Filter statistics by range in shot clock

last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

num_contested_rebounding()

`rebound_distance_rebounding()`

`shot_distance_rebounding()`

`shot_type_rebounding()`

```
class nba_py.player.PlayerShootingSplits (player_id, team_id=0, measure_type='Base',
                                           per_mode='PerGame', plus_minus='N',
                                           pace_adjust='N', rank='N', league_id='00',
                                           season='2017-18', season_type='Regular Season',
                                           po_round='0', outcome='', location='', month='0',
                                           season_segment='', date_from='', date_to='',
                                           opponent_team_id='0', vs_conference='',
                                           vs_division='', game_segment='', period='0',
                                           shot_clock_range='', last_n_games='0')
```

Bases: `nba_py.player._PlayerDashboard`

Shooting stats based on distance, area, assisted to, shot types, and assisted by.

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division

game_segment Filter by half / overtime

period Filter by quarter / specific overtime

shot_clock_range Filter statistics by range in shot clock

last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

assisted_by ()

assisted_shots ()

shot_5ft ()

shot_8ft ()

shot_areas ()

shot_types_detail ()

shot_types_summary ()

```
class nba_py.player.PlayerShotLogTracking (player_id, team_id=0, measure_type='Base',
                                           per_mode='PerGame', plus_minus='N',
                                           pace_adjust='N', rank='N', league_id='00',
                                           season='2017-18', season_type='Regular
Season', po_round='0', outcome='', lo-
cation='', month='0', season_segment='',
date_from='', date_to='', oppo-
nent_team_id='0', vs_conference='',
vs_division='', game_segment='', period='0',
shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Contains a log for every shot for a given season for a given player

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division

game_segment Filter by half / overtime

period Filter by quarter / specific overtime

shot_clock_range Filter statistics by range in shot clock

last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

```
class nba_py.player.PlayerShotTracking (player_id, team_id=0, measure_type='Base',
                                       per_mode='PerGame', plus_minus='N',
                                       pace_adjust='N', rank='N', league_id='00',
                                       season='2017-18', season_type='Regular Season',
                                       po_round='0', outcome='', location='', month='0',
                                       season_segment='', date_from='', date_to='', oppo-
                                       nent_team_id='0', vs_conference='', vs_division='',
                                       game_segment='', period='0', shot_clock_range='',
                                       last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Tracking data for shooting for a given player

Args:

player_id ID of the player to look up

team_id ID of the team to look up

measure_type Specifies type of measure to use (Base, Advanced, etc.)

per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)

plus_minus Whether or not to consider plus minus (Y or N)

pace_adjust Whether or not to pace adjust stats (Y or N)

rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)

season Season given to look up

season_type Season type to consider (Regular / Playoffs)

po_round Playoff round

outcome Filter out by wins or losses

location Filter out by home or away

month Specify month to filter by

season_segment Filter by pre/post all star break

date_from Filter out games before a specific date

date_to Filter out games after a specific date

opponent_team_id Opponent team ID to look up

vs_conference Filter by conference

vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

closest_defender_shooting ()
closest_defender_shooting_long ()
dribble_shooting ()
general_shooting ()
shot_clock_shooting ()
touch_time_shooting ()

class nba_py.player.**PlayerSummary** (*player_id*)
 Contains common player information like headline stats, weight, etc.

Args:

player_id ID of the player to look up

Attributes:

json Contains the full json dump to play around with

headline_stats ()
info ()

class nba_py.player.**PlayerVsPlayer** (*player_id*, *vs_player_id*, *team_id=0*, *measure_type='Base'*, *per_mode='PerGame'*, *plus_minus='N'*, *pace_adjust='N'*, *rank='N'*, *league_id='00'*, *season='2017-18'*, *season_type='Regular Season'*, *po_round='0'*, *outcome=''*, *location=''*, *month='0'*, *season_segment=''*, *date_from=''*, *date_to=''*, *opponent_team_id='0'*, *vs_conference=''*, *vs_division=''*, *game_segment=''*, *period='0'*, *shot_clock_range=''*, *last_n_games='0'*)

Contains general stats that pertain to players going against other players

Args:

player_id ID of the player to look up
vs_player_id ID of the vs player to look up
team_id ID of the team to look up
measure_type Specifies type of measure to use (Base, Advanced, etc.)
per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)
plus_minus Whether or not to consider plus minus (Y or N)
pace_adjust Whether or not to pace adjust stats (Y or N)
rank Whether or not to consider rank (Y or N)

league_id ID for the league to look in (Default is 00)
season Season given to look up
season_type Season type to consider (Regular / Playoffs)
po_round Playoff round
outcome Filter out by wins or losses
location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes: json: Contains the full json dump to play around with

on_off_court ()

overall ()

player_info ()

shot_area_off_court ()

shot_area_on_court ()

shot_area_overall ()

shot_distance_off_court ()

shot_distance_on_court ()

shot_distance_overall ()

vs_player_info ()

```
class nba_py.player.PlayerYearOverYearSplits (player_id, team_id=0, measure_type='Base',
                                              per_mode='PerGame', plus_minus='N',
                                              pace_adjust='N', rank='N', league_id='00',
                                              season='2017-18', season_type='Regular
Season', po_round='0', outcome='', loca-
tion='', month='0', season_segment='',
date_from='', date_to='', oppo-
nent_team_id='0', vs_conference='',
vs_division='', game_segment='', period='0',
shot_clock_range='', last_n_games='0')
```

Bases: nba_py.player._PlayerDashboard

Displays player stats over the given season and over all seasons in the given league

Args:

player_id ID of the player to look up
team_id ID of the team to look up
measure_type Specifies type of measure to use (Base, Advanced, etc.)
per_mode Mode to measure statistics (Totals, PerGame, Per36, etc.)
plus_minus Whether or not to consider plus minus (Y or N)
pace_adjust Whether or not to pace adjust stats (Y or N)
rank Whether or not to consider rank (Y or N)
league_id ID for the league to look in (Default is 00)
season Season given to look up
season_type Season type to consider (Regular / Playoffs)
po_round Playoff round
outcome Filter out by wins or losses
location Filter out by home or away
month Specify month to filter by
season_segment Filter by pre/post all star break
date_from Filter out games before a specific date
date_to Filter out games after a specific date
opponent_team_id Opponent team ID to look up
vs_conference Filter by conference
vs_division Filter by division
game_segment Filter by half / overtime
period Filter by quarter / specific overtime
shot_clock_range Filter statistics by range in shot clock
last_n_games Filter by number of games specified in N

Attributes:

json Contains the full json dump to play around with

by_year ()

```
nba_py.player.get_player(first_name, last_name=None, season='2017-18', only_current=0,  
                        just_id=True)
```

Calls our PlayerList class to get a full list of players and then returns just an id if specified or the full row of player information

Args:

first_name First name of the player

last_name Last name of the player

(this is None if the player only has first name [None]) :only_current: Only wants the current list of players

:just_id: Only wants the id of the player

Returns: Either the ID or full row of information of the player inputted

Raises: :PlayerNotFoundException:

nba_py.game module

```
class nba_py.game.Boxscore(game_id, season='2017-18', season_type='Regular Season',
                             range_type='0', start_period='0', end_period='0', start_range='0',
                             end_range='0')
    Bases: nba_py.game._BaseBoxcore
    player_stats()
    team_starter_bench_stats()
    team_stats()

class nba_py.game.BoxscoreAdvanced(game_id, season='2017-18', season_type='Regular Season',
                                     range_type='0', start_period='0', end_period='0',
                                     start_range='0', end_range='0')
    Bases: nba_py.game._BaseBoxcore
    sql_players_advanced()
    sql_team_advanced()

class nba_py.game.BoxscoreFourFactors(game_id, season='2017-18', season_type='Regular Season',
                                         range_type='0', start_period='0', end_period='0',
                                         start_range='0', end_range='0')
    Bases: nba_py.game._BaseBoxcore
    sql_players_four_factors()
    sql_team_four_factors()

class nba_py.game.BoxscoreMisc(game_id, season='2017-18', season_type='Regular Season',
                                  range_type='0', start_period='0', end_period='0',
                                  start_range='0', end_range='0')
    Bases: nba_py.game._BaseBoxcore
    sql_players_misc()
    sql_team_misc()
```

```
class nba_py.game.BoxscoreScoring(game_id, season='2017-18', season_type='Regular Season', range_type='0', start_period='0', end_period='0', start_range='0', end_range='0')
```

```
    Bases: nba_py.game._BaseBoxcore
```

```
    sql_players_scoring()
```

```
    sql_team_scoring()
```

```
class nba_py.game.BoxscoreSummary(game_id, season='2017-18', season_type='Regular Season', range_type='0', start_period='0', end_period='0', start_range='0', end_range='0')
```

```
    available_video()
```

```
    game_info()
```

```
    game_summary()
```

```
    inactive_players()
```

```
    last_meeting()
```

```
    line_score()
```

```
    officials()
```

```
    other_stats()
```

```
    season_series()
```

```
class nba_py.game.BoxscoreUsage(game_id, season='2017-18', season_type='Regular Season', range_type='0', start_period='0', end_period='0', start_range='0', end_range='0')
```

```
    Bases: nba_py.game._BaseBoxcore
```

```
    sql_players_usage()
```

```
    sql_team_usage()
```

```
class nba_py.game.HustleStats(game_id)
```

```
    hustle_stats_available()
```

```
    hustle_stats_player_box_score()
```

```
    hustle_stats_team_box_score()
```

```
class nba_py.game.PlayByPlay(game_id, start_period='0', end_period='0')
```

```
    available_video()
```

```
    info()
```

```
class nba_py.game.PlayerTracking(game_id)
```

```
    info()
```

nba_py.team module

```
class nba_py.team.TeamClutchSplits (team_id, measure_type='Base', per_mode='PerGame',
                                   plus_minus='N', pace_adjust='N', rank='N',
                                   league_id='00', season='2017-18', season_type='Regular
                                   Season', po_round='0', outcome='', location='', month='0',
                                   season_segment='', date_from='', date_to='', oppo-
                                   nent_team_id='0', vs_conference='', vs_division='',
                                   game_segment='', period='0', shot_clock_range='',
                                   last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

This is a weird endpoint, to be honest. It's got a lot of cool little stats and there are two extra fields in the json that I have no idea what they do.

If you know please tell me.

- Last30Sec3Point2TeamDashboard
- Last10Sec3Point2TeamDashboard

last10sec_deficit_3point ()
Results in last 5 minutes <= 5 points

last1min_deficit_5point ()
Results in last 5 minutes <= 5 points

last1min_plusminus_5point ()
Last 1 minutes +/- 5 points

last30sec_deficit_3point ()
Results in last 5 minutes <= 5 points

last30sec_plusminus_5point ()
Last 30 seconds +/- 3 points

last3min_deficit_5point ()
Results in last 5 minutes <= 5 points

last3min_plusminus_5point ()

Last 3 minutes +/- 5 points

last5min_deficit_5point ()

Results in last 5 minutes <= 5 points

last5min_plusminus_5point ()

Last 5 minutes +/- 5 points

class nba_py.team.**TeamCommonRoster** (*team_id*, *season*='2017-18')

coaches ()

roster ()

class nba_py.team.**TeamDetails** (*team_id*)

awards_championships ()

awards_conf ()

awards_div ()

background ()

history ()

hof ()

retired ()

social_sites ()

class nba_py.team.**TeamGameLogs** (*team_id*, *season*='2017-18', *season_type*='Regular Season')

info ()

class nba_py.team.**TeamGeneralSplits** (*team_id*, *measure_type*='Base', *per_mode*='PerGame',
plus_minus='N', *pace_adjust*='N', *rank*='N',
league_id='00', *season*='2017-18', *season_type*='Regular
Season', *po_round*='0', *outcome*='', *location*='',
month='0', *season_segment*='', *date_from*='', *date_to*='',
opponent_team_id='0', *vs_conference*='', *vs_division*='',
game_segment='', *period*='0', *shot_clock_range*='',
last_n_games='0')

Bases: nba_py.team._TeamDashboard

days_rest ()

location ()

monthly ()

pre_post_all_star ()

wins_losses ()


```
class nba_py.team.TeamInGameSplits (team_id, measure_type='Base', per_mode='PerGame',
                                     plus_minus='N', pace_adjust='N', rank='N',
                                     league_id='00', season='2017-18', season_type='Regular
                                     Season', po_round='0', outcome='', location='', month='0',
                                     season_segment='', date_from='', date_to='', oppo-
                                     nent_team_id='0', vs_conference='', vs_division='',
                                     game_segment='', period='0', shot_clock_range='',
                                     last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

by_actual_margin()

by_half()

by_period()

by_score_margin()

```
class nba_py.team.TeamLastNGamesSplits (team_id, measure_type='Base', per_mode='PerGame',
                                          plus_minus='N', pace_adjust='N', rank='N',
                                          league_id='00', season='2017-18', sea-
                                          son_type='Regular Season', po_round='0', out-
                                          come='', location='', month='0', season_segment='',
                                          date_from='', date_to='', opponent_team_id='0',
                                          vs_conference='', vs_division='', game_segment='',
                                          period='0', shot_clock_range='', last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

gamenumber()

last10()

last15()

last20()

last5()

```
class nba_py.team.TeamLineups (team_id, game_id='', group_quantity=5, season='2017-
                               18', season_type='Regular Season', measure_type='Base',
                               per_mode='PerGame', plus_minus='N', pace_adjust='N',
                               rank='N', outcome='', location='', month='0', season_segment='',
                               date_from='', date_to='', opponent_team_id='0', vs_conference='',
                               vs_division='', game_segment='', period='0', last_n_games='0')
```

lineups()

overall()

```
class nba_py.team.TeamList (league_id='00')
```

info()

```
class nba_py.team.TeamOpponentSplits (team_id, measure_type='Base', per_mode='PerGame',
                                       plus_minus='N', pace_adjust='N', rank='N',
                                       league_id='00', season='2017-18', sea-
                                       son_type='Regular Season', po_round='0', out-
                                       come='', location='', month='0', season_segment='',
                                       date_from='', date_to='', opponent_team_id='0',
                                       vs_conference='', vs_division='', game_segment='',
                                       period='0', shot_clock_range='', last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

by_conference ()

by_division ()

by_opponent ()

```
class nba_py.team.TeamPassTracking (team_id, measure_type='Base', per_mode='PerGame',
                                   plus_minus='N', pace_adjust='N', rank='N',
                                   league_id='00', season='2017-18', season_type='Regular
                                   Season', po_round='0', outcome='', location='', month='0',
                                   season_segment='', date_from='', date_to='', oppo-
                                   nent_team_id='0', vs_conference='', vs_division='',
                                   game_segment='', period='0', shot_clock_range='',
                                   last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

passes_made ()

passes_recieved ()

```
class nba_py.team.TeamPerformanceSplits (team_id, measure_type='Base',
                                          per_mode='PerGame', plus_minus='N',
                                          pace_adjust='N', rank='N', league_id='00',
                                          season='2017-18', season_type='Regular Season',
                                          po_round='0', outcome='', location='', month='0',
                                          season_segment='', date_from='', date_to='', oppo-
                                          nent_team_id='0', vs_conference='', vs_division='',
                                          game_segment='', period='0', shot_clock_range='',
                                          last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

points_against ()

points_scored ()

score_differential ()

```
class nba_py.team.TeamPlayerOnOffDetail (team_id, measure_type='Base',
                                          per_mode='PerGame', plus_minus='N',
                                          pace_adjust='N', rank='N', league_id='00',
                                          season='2017-18', season_type='Regular Season',
                                          po_round='0', outcome='', location='', month='0',
                                          season_segment='', date_from='', date_to='', oppo-
                                          nent_team_id='0', vs_conference='', vs_division='',
                                          game_segment='', period='0', shot_clock_range='',
                                          last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

off_court ()

on_court ()

```
class nba_py.team.TeamPlayerOnOffSummary (team_id,
                                         measure_type='Base',
                                         per_mode='PerGame',
                                         plus_minus='N',
                                         pace_adjust='N',
                                         rank='N',
                                         league_id='00',
                                         season='2017-18',
                                         season_type='Regular Season',
                                         po_round='0',
                                         outcome='',
                                         location='',
                                         month='0',
                                         season_segment='',
                                         date_from='',
                                         date_to='',
                                         opponent_team_id='0',
                                         vs_conference='',
                                         vs_division='',
                                         game_segment='',
                                         period='0',
                                         shot_clock_range='',
                                         last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

off_court ()

on_court ()

```
class nba_py.team.TeamPlayers (team_id,
                               measure_type='Base',
                               per_mode='PerGame',
                               plus_minus='N',
                               pace_adjust='N',
                               rank='N',
                               league_id='00',
                               season='2017-18',
                               season_type='Regular Season',
                               po_round='0',
                               outcome='',
                               location='',
                               month='0',
                               season_segment='',
                               date_from='',
                               date_to='',
                               opponent_team_id='0',
                               vs_conference='',
                               vs_division='',
                               game_segment='',
                               period='0',
                               shot_clock_range='',
                               last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

season_totals ()

```
class nba_py.team.TeamReboundTracking (team_id,
                                       measure_type='Base',
                                       per_mode='PerGame',
                                       plus_minus='N',
                                       pace_adjust='N',
                                       rank='N',
                                       league_id='00',
                                       season='2017-18',
                                       season_type='Regular Season',
                                       po_round='0',
                                       outcome='',
                                       location='',
                                       month='0',
                                       season_segment='',
                                       date_from='',
                                       date_to='',
                                       opponent_team_id='0',
                                       vs_conference='',
                                       vs_division='',
                                       game_segment='',
                                       period='0',
                                       shot_clock_range='',
                                       last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

contested_rebounding ()

rebound_distance_rebounding ()

shot_distance_rebounding ()

shot_type_rebounding ()

```
class nba_py.team.TeamSeasons (team_id,
                               league_id='00',
                               season_type='Regular Season',
                               per_mode='PerGame')
```

info ()

```
class nba_py.team.TeamShootingSplits (team_id,
                                       measure_type='Base',
                                       per_mode='PerGame',
                                       plus_minus='N',
                                       pace_adjust='N',
                                       rank='N',
                                       league_id='00',
                                       season='2017-18',
                                       season_type='Regular Season',
                                       po_round='0',
                                       outcome='',
                                       location='',
                                       month='0',
                                       season_segment='',
                                       date_from='',
                                       date_to='',
                                       opponent_team_id='0',
                                       vs_conference='',
                                       vs_division='',
                                       game_segment='',
                                       period='0',
                                       shot_clock_range='',
                                       last_n_games='0')
```

Bases: nba_py.team._TeamDashboard

assisted_by ()

```
assisted_shots ()
shot_5ft ()
shot_8ft ()
shot_areas ()
shot_type_summary ()
class nba_py.team.TeamShotTracking (team_id, measure_type='Base', per_mode='PerGame',
plus_minus='N', pace_adjust='N', rank='N',
league_id='00', season='2017-18', season_type='Regular
Season', po_round='0', outcome='', location='', month='0',
season_segment='', date_from='', date_to='', oppo-
nent_team_id='0', vs_conference='', vs_division='',
game_segment='', period='0', shot_clock_range='',
last_n_games='0')
Bases: nba_py.team._TeamDashboard
closest_defender_shooting ()
closest_defender_shooting_long ()
dribble_shooting ()
shot_clock_shooting ()
touch_time_shooting ()
class nba_py.team.TeamSummary (team_id, season='2017-18', league_id='00', season_type='Regular
Season')
info ()
season_ranks ()
class nba_py.team.TeamVsPlayer (team_id, vs_player_id, measure_type='Base',
per_mode='PerGame', plus_minus='N', pace_adjust='N',
rank='N', league_id='00', season='2017-18', sea-
son_type='Regular Season', po_round='0', outcome='', loca-
tion='', month='0', season_segment='', date_from='', date_to='',
opponent_team_id='0', vs_conference='', vs_division='',
game_segment='', period='0', shot_clock_range='',
last_n_games='0')
on_off_court ()
overall ()
shot_area_off_court ()
shot_area_on_court ()
shot_area_overall ()
shot_distance_off_court ()
shot_distance_on_court ()
shot_distance_overall ()
vs_player_overall ()
```

```
class nba_py.team.TeamYearOverYearSplits (team_id, measure_type='Base',  
                                           per_mode='PerGame', plus_minus='N',  
                                           pace_adjust='N', rank='N', league_id='00',  
                                           season='2017-18', season_type='Regular Season',  
                                           po_round='0', outcome='', location='', month='0',  
                                           season_segment='', date_from='', date_to='',  
                                           opponent_team_id='0', vs_conference='',  
                                           vs_division='', game_segment='', period='0',  
                                           shot_clock_range='', last_n_games='0')  
  
Bases: nba_py.team._TeamDashboard  
by_year ()
```



```
class nba_py.constants.AheadBehind
    Bases: nba_py.constants._DefaultBlank
    AheadOrBehind = 'Ahead or Behind'
    AheadOrTied = 'Ahead or Tied'
    BehindOrTied = 'Behind or Tied'
class nba_py.constants.ClutchTime
    Bases: nba_py.constants._DefaultBlank
    Last10Sec = 'Last 10 Seconds'
    Last1Min = 'Last 1 Minutes'
    Last2Min = 'Last 2 Minutes'
    Last30Sec = 'Last 30 Seconds'
    Last3Min = 'Last 3 Minutes'
    Last4Min = 'Last 4 Minutes'
    Last5Min = 'Last 5 Minutes'
class nba_py.constants.College
    Bases: nba_py.constants._DefaultBlank
class nba_py.constants.Conference
    Bases: nba_py.constants.VsConference
class nba_py.constants.ContextMeasure
    Default = 'FGM'
    EFG_PCT = 'EFG_PCT'
    FG3A = 'FG3A'
```

```
FG3M = 'FG3m'  
FG3_PCT = 'FG3_PCT'  
FGA = 'FGA'  
FGM = 'FGM'  
FG_PCT = 'FG_PCT'  
PF = 'PF'  
PTS_2ND_CHANCE = 'PTS_2ND_CHANCE'  
PTS_FB = 'PTS_FB'  
PTS_OFF_TOV = 'PTS_OFF_TOV'  
TS_PCT = 'TS_PCT'
```

```
class nba_py.constants.Counter
```

```
    Default = '1000'
```

```
class nba_py.constants.Country  
    Bases: nba_py.constants._DefaultBlank
```

```
class nba_py.constants.DateFrom  
    Bases: nba_py.constants._DefaultBlank
```

```
class nba_py.constants.DateTo  
    Bases: nba_py.constants._DefaultBlank
```

```
class nba_py.constants.Direction
```

```
    ASC = 'ASC'
```

```
    DESC = 'DESC'
```

```
    Default = 'DESC'
```

```
class nba_py.constants.Division  
    Bases: nba_py.constants.VsDivision
```

```
class nba_py.constants.DraftPick  
    Bases: nba_py.constants._DefaultBlank
```

```
    FirstPick = '1st+Pick'
```

```
    FirstRound = '1st+Round'
```

```
    Lottery = 'Lottery+Pick'
```

```
    Picks11Thru20 = 'Picks+11+Thru+20'
```

```
    Picks21Thru30 = 'Picks+21+Thru+30'
```

```
    SecondRound = '2nd+Round'
```

```
    Top10 = 'Top+10+Pick'
```

```
    Top15 = 'Top+15+Pick'
```

```
    Top20 = 'Top+20+Pick'
```

```
    Top25 = 'Top+25+Pick'
```


Top5 = 'Top+5+Pick'

Undrafted = 'Undrafted'

class nba_py.constants.**DraftYear**
Bases: nba_py.constants._DefaultBlank

class nba_py.constants.**EndPeriod**
Bases: *nba_py.constants.Period*

class nba_py.constants.**EndRange**
Bases: nba_py.constants._DefaultZero

class nba_py.constants.**GameID**
Bases: nba_py.constants._DefaultBlank

class nba_py.constants.**GameScope**

Default = 'Season'

Finals = 'Finals'

Last10 = 'Last 10'

Season = 'Season'

Yesterday = 'Yesterday'

class nba_py.constants.**GameSegment**
Bases: nba_py.constants._DefaultBlank

EntireGame = ''

FirstHalf = 'First Half'

Overtime = 'Overtime'

SecondHalf = 'Second Half'

class nba_py.constants.**Game_Scope**
Bases: nba_py.constants._DefaultBlank

Last10 = 'Last 10'

Yesterday = 'Yesterday'

class nba_py.constants.**GroupQuantity**

Default = 5

class nba_py.constants.**Height**
Bases: nba_py.constants._DefaultBlank

Example: for greater than 6ft8 api call should be GT+6-8 for lower than 7ft3 api call should be LT+7-3

class nba_py.constants.**LastNGames**
Bases: nba_py.constants._DefaultZero

class nba_py.constants.**League**

Default = '00'

NBA = '00'

```
class nba_py.constants.Location
    Bases: nba_py.constants._DefaultBlank
```

```
    Away = 'Away'
```

```
    Home = 'Home'
```

```
class nba_py.constants.MeasureType
```

```
    Advanced = 'Advanced'
```

```
    Base = 'Base'
```

```
    Default = 'Base'
```

```
    FourFactors = 'Four Factors'
```

```
    Misc = 'Misc'
```

```
    Opponent = 'Opponent'
```

```
    Scoring = 'Scoring'
```

```
    Usage = 'Usage'
```

```
class nba_py.constants.Month
    Bases: nba_py.constants._DefaultZero
```

```
    All = '0'
```

```
    April = '7'
```

```
    August = '11'
```

```
    December = '3'
```

```
    February = '5'
```

```
    January = '4'
```

```
    July = '10'
```

```
    June = '9'
```

```
    March = '6'
```

```
    May = '8'
```

```
    November = '2'
```

```
    October = '1'
```

```
    September = '12'
```

```
class nba_py.constants.OpponentTeamID
    Bases: nba_py.constants._DefaultZero
```

```
class nba_py.constants.Outcome
    Bases: nba_py.constants._DefaultBlank
```

```
    Loss = 'L'
```

```
    Win = 'W'
```

```
class nba_py.constants.PaceAdjust
    Bases: nba_py.constants._DefaultN
```

```
class nba_py.constants.PerMode
```

```
    Default = 'PerGame'
    MinutesPer = 'MinutesPer'
    Per100Plays = 'Per100Plays'
    Per100Possessions = 'Per100Possessions'
    Per36 = 'Per36'
    Per40 = 'Per40'
    Per48 = 'Per48'
    PerGame = 'PerGame'
    PerMinute = 'PerMinute'
    PerPlay = 'PerPlay'
    PerPossession = 'PerPossession'
    Totals = 'Totals'
```

```
class nba_py.constants.Period
    Bases: nba_py.constants._DefaultZero
```

```
    AllQuarters = '0'
    FirstQuarter = '1'
    FourthQuarter = '4'
    Overtime (n)
    SecondQuarter = '2'
    ThirdQuarter = '3'
```

```
class nba_py.constants.PlayerExperience
    Bases: nba_py.constants._DefaultBlank
```

```
    Rookie = 'Rookie'
    Sophomore = 'Sophomore'
    Veteran = 'Veteran'
```

```
class nba_py.constants.PlayerOrTeam
```

```
    Default = 'Player'
    Player = 'Player'
    Team = 'Team'
```

```
class nba_py.constants.PlayerPosition
    Bases: nba_py.constants._DefaultBlank
```

```
    Center = 'C'
    Forward = 'F'
    Guard = 'G'
```

```
class nba_py.constants.PlayerScope

    AllPlayers = 'All Players'
    Default = 'All Players'
    Rookies = 'Rookie'

class nba_py.constants.Player_or_Team

    Default = 'P'
    Player = 'P'
    Team = 'T'

class nba_py.constants.PlayoffRound
    Bases: nba_py.constants._DefaultZero

    All = '0'
    ConferenceFinals = '3'
    Finals = '4'
    QuarterFinals = '1'
    SemiFinals = '2'

class nba_py.constants.PlusMinus
    Bases: nba_py.constants._DefaultN

class nba_py.constants.PtMeasureType

    SpeedDistance = 'SpeedDistance'

class nba_py.constants.RangeType
    Bases: nba_py.constants._DefaultZero

class nba_py.constants.Rank
    Bases: nba_py.constants._DefaultN

class nba_py.constants.RookieYear
    Bases: nba_py.constants._DefaultBlank

class nba_py.constants.Scope

    AllPlayers = 'S'
    Default = 'S'
    Rookies = 'Rookies'

class nba_py.constants.SeasonSegment
    Bases: nba_py.constants._DefaultBlank

    EntireSeason = ''
    PostAllStar = 'Post All-Star'
    PreAllStar = 'Pre All-Star'

class nba_py.constants.SeasonType
```

```
    Default = 'Regular Season'
    Playoffs = 'Playoffs'
    Regular = 'Regular Season'
class nba_py.constants.ShotClockRange
    Bases: nba_py.constants._DefaultBlank
    AllRanges = ''
    ShotClockOff = 'ShotClock Off'
    get (n)
class nba_py.constants.Sorter

    AST = 'AST'
    BLK = 'BLK'
    DREB = 'DREB'
    Default = 'PTS'
    FG3A = 'FG3A'
    FG3M = 'FG3M'
    FG3_PCT = 'FG3_PCT'
    FGA = 'FGA'
    FGM = 'FGM'
    FG_PCT = 'FG_PCT'
    FTA = 'FTA'
    FTM = 'FTM'
    FT_PCT = 'FT_PCT'
    OREB = 'OREB'
    PTS = 'PTS'
    REB = 'REB'
    STL = 'STL'
    TOV = 'TOV'
class nba_py.constants.StartPeriod
    Bases: nba_py.constants.Period
class nba_py.constants.StartRange
    Bases: nba_py.constants._DefaultZero
class nba_py.constants.StarterBench
    Bases: nba_py.constants._DefaultBlank
    Bench = 'Bench'
    Starters = 'Starters'
class nba_py.constants.StatCategory
```

```
AST = 'AST'  
AST_TOV = 'AST/TO'  
BLK = 'BLK'  
DREB = 'DREB'  
Default = 'PTS'  
EFF = 'EFF'  
FG3A = '3PA'  
FG3M = '3PM'  
FG3_PCT = '3P%'  
FGA = 'FGA'  
FGM = 'FGM'  
FG_PCT = 'FG%'  
FTA = 'FTA'  
FTM = 'FTM'  
FT_PCT = 'FT%'  
OREB = 'OREB'  
PF = 'PF'  
PTS = 'PTS'  
REB = 'REB'  
STL = 'STL'  
STL_TOV = 'STL/TOV'  
TOV = 'TOV'
```

```
class nba_py.constants.TeamID  
    Bases: nba_py.constants._DefaultZero  
  
class nba_py.constants.VsConference  
    Bases: nba_py.constants._DefaultBlank  
  
    All = ''  
  
    East = 'East'  
  
    West = 'West'  
  
class nba_py.constants.VsDivision  
    Bases: nba_py.constants._DefaultBlank  
  
    All = ''  
  
    Atlantic = 'Atlantic'  
  
    Central = 'Central'  
  
    Northwest = 'Northwest'  
  
    Pacific = 'Pacific'  
  
    Southeast = 'Southeast'
```

Southwest = 'Southwest'

class nba_py.constants.**Weight**

Bases: nba_py.constants._DefaultBlank

Example: for greater than 225lbs api call should be GT+225lbs

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

n

nba_py, 3
nba_py.constants, 35
nba_py.game, 25
nba_py.player, 5
nba_py.team, 27

A

Advanced (nba_py.constants.MeasureType attribute), 38
 AheadBehind (class in nba_py.constants), 35
 AheadOrBehind (nba_py.constants.AheadBehind attribute), 35
 AheadOrTied (nba_py.constants.AheadBehind attribute), 35
 All (nba_py.constants.Month attribute), 38
 All (nba_py.constants.PlayoffRound attribute), 40
 All (nba_py.constants.VsConference attribute), 42
 All (nba_py.constants.VsDivision attribute), 42
 all_star_season_totals() (nba_py.player.PlayerCareer method), 5
 AllPlayers (nba_py.constants.PlayerScope attribute), 40
 AllPlayers (nba_py.constants.Scope attribute), 40
 AllQuarters (nba_py.constants.Period attribute), 39
 AllRanges (nba_py.constants.ShotClockRange attribute), 41
 April (nba_py.constants.Month attribute), 38
 ASC (nba_py.constants.Direction attribute), 36
 assisted_by() (nba_py.player.PlayerShootingSplits method), 19
 assisted_by() (nba_py.team.TeamShootingSplits method), 31
 assisted_shots() (nba_py.player.PlayerShootingSplits method), 19
 assisted_shots() (nba_py.team.TeamShootingSplits method), 31
 AST (nba_py.constants.Sorter attribute), 41
 AST (nba_py.constants.StatCategory attribute), 41
 AST_TOV (nba_py.constants.StatCategory attribute), 42
 Atlantic (nba_py.constants.VsDivision attribute), 42
 August (nba_py.constants.Month attribute), 38
 available() (nba_py.Scoreboard method), 3
 available_video() (nba_py.game.BoxscoreSummary method), 26
 available_video() (nba_py.game.PlayByPlay method), 26
 awards_championships() (nba_py.team.TeamDetails method), 28

awards_conf() (nba_py.team.TeamDetails method), 28
 awards_div() (nba_py.team.TeamDetails method), 28
 Away (nba_py.constants.Location attribute), 38

B

background() (nba_py.team.TeamDetails method), 28
 Base (nba_py.constants.MeasureType attribute), 38
 BehindOrTied (nba_py.constants.AheadBehind attribute), 35
 Bench (nba_py.constants.StarterBench attribute), 41
 BLK (nba_py.constants.Sorter attribute), 41
 BLK (nba_py.constants.StatCategory attribute), 42
 Boxscore (class in nba_py.game), 25
 BoxscoreAdvanced (class in nba_py.game), 25
 BoxscoreFourFactors (class in nba_py.game), 25
 BoxscoreMisc (class in nba_py.game), 25
 BoxscoreScoring (class in nba_py.game), 25
 BoxscoreSummary (class in nba_py.game), 26
 BoxscoreUsage (class in nba_py.game), 26
 by_actual_margin() (nba_py.player.PlayerInGameSplits method), 10
 by_actual_margin() (nba_py.team.TeamInGameSplits method), 29
 by_conference() (nba_py.player.PlayerOpponentSplits method), 13
 by_conference() (nba_py.team.TeamOpponentSplits method), 30
 by_division() (nba_py.player.PlayerOpponentSplits method), 13
 by_division() (nba_py.team.TeamOpponentSplits method), 30
 by_half() (nba_py.player.PlayerInGameSplits method), 10
 by_half() (nba_py.team.TeamInGameSplits method), 29
 by_opponent() (nba_py.player.PlayerOpponentSplits method), 13
 by_opponent() (nba_py.team.TeamOpponentSplits method), 30
 by_period() (nba_py.player.PlayerInGameSplits method), 10

by_period() (nba_py.team.TeamInGameSplits method), 29
 by_score_margin() (nba_py.player.PlayerInGameSplits method), 10
 by_score_margin() (nba_py.team.TeamInGameSplits method), 29
 by_year() (nba_py.player.PlayerYearOverYearSplits method), 23
 by_year() (nba_py.team.TeamYearOverYearSplits method), 33

C

career_all_star_season_totals() (nba_py.player.PlayerCareer method), 5
 career_highs() (nba_py.player.PlayerProfile method), 15
 Center (nba_py.constants.PlayerPosition attribute), 39
 Central (nba_py.constants.VsDivision attribute), 42
 closest_defender_shooting() (nba_py.player.PlayerShotTracking method), 21
 closest_defender_shooting() (nba_py.team.TeamShotTracking method), 32
 closest_defender_shooting_long() (nba_py.player.PlayerShotTracking method), 21
 closest_defender_shooting_long() (nba_py.team.TeamShotTracking method), 32
 ClutchTime (class in nba_py.constants), 35
 coaches() (nba_py.team.TeamCommonRoster method), 28
 College (class in nba_py.constants), 35
 college_season_career_totals() (nba_py.player.PlayerCareer method), 5
 college_season_totals() (nba_py.player.PlayerCareer method), 5
 Conference (class in nba_py.constants), 35
 ConferenceFinals (nba_py.constants.PlayoffRound attribute), 40
 contested_rebounding() (nba_py.team.TeamReboundTracking method), 31
 ContextMeasure (class in nba_py.constants), 35
 Counter (class in nba_py.constants), 36
 Country (class in nba_py.constants), 36

D

DateFrom (class in nba_py.constants), 36
 DateTo (class in nba_py.constants), 36
 days_rest() (nba_py.player.PlayerGeneralSplits method), 9
 days_rest() (nba_py.team.TeamGeneralSplits method), 28
 December (nba_py.constants.Month attribute), 38
 Default (nba_py.constants.ContextMeasure attribute), 35

Default (nba_py.constants.Counter attribute), 36
 Default (nba_py.constants.Direction attribute), 36
 Default (nba_py.constants.GameScope attribute), 37
 Default (nba_py.constants.GroupQuantity attribute), 37
 Default (nba_py.constants.League attribute), 37
 Default (nba_py.constants.MeasureType attribute), 38
 Default (nba_py.constants.PerMode attribute), 39
 Default (nba_py.constants.Player_or_Team attribute), 40
 Default (nba_py.constants.PlayerOrTeam attribute), 39
 Default (nba_py.constants.PlayerScope attribute), 40
 Default (nba_py.constants.Scope attribute), 40
 Default (nba_py.constants.SeasonType attribute), 40
 Default (nba_py.constants.Sorter attribute), 41
 Default (nba_py.constants.StatCategory attribute), 42
 DESC (nba_py.constants.Direction attribute), 36
 Direction (class in nba_py.constants), 36
 Division (class in nba_py.constants), 36
 DraftPick (class in nba_py.constants), 36
 DraftYear (class in nba_py.constants), 37
 DREB (nba_py.constants.Sorter attribute), 41
 DREB (nba_py.constants.StatCategory attribute), 42
 dribble_shooting() (nba_py.player.PlayerShotTracking method), 21
 dribble_shooting() (nba_py.team.TeamShotTracking method), 32

E

East (nba_py.constants.VsConference attribute), 42
 east_conf_standings_by_day() (nba_py.Scoreboard method), 3
 EFF (nba_py.constants.StatCategory attribute), 42
 EFG_PCT (nba_py.constants.ContextMeasure attribute), 35
 EndPeriod (class in nba_py.constants), 37
 EndRange (class in nba_py.constants), 37
 EntireGame (nba_py.constants.GameSegment attribute), 37
 EntireSeason (nba_py.constants.SeasonSegment attribute), 40

F

February (nba_py.constants.Month attribute), 38
 FG3_PCT (nba_py.constants.ContextMeasure attribute), 36
 FG3_PCT (nba_py.constants.Sorter attribute), 41
 FG3_PCT (nba_py.constants.StatCategory attribute), 42
 FG3A (nba_py.constants.ContextMeasure attribute), 35
 FG3A (nba_py.constants.Sorter attribute), 41
 FG3A (nba_py.constants.StatCategory attribute), 42
 FG3M (nba_py.constants.ContextMeasure attribute), 35
 FG3M (nba_py.constants.Sorter attribute), 41
 FG3M (nba_py.constants.StatCategory attribute), 42
 FG_PCT (nba_py.constants.ContextMeasure attribute), 36

- FG_PCT (nba_py.constants.Sorter attribute), 41
 FG_PCT (nba_py.constants.StatCategory attribute), 42
 FGA (nba_py.constants.ContextMeasure attribute), 36
 FGA (nba_py.constants.Sorter attribute), 41
 FGA (nba_py.constants.StatCategory attribute), 42
 FGM (nba_py.constants.ContextMeasure attribute), 36
 FGM (nba_py.constants.Sorter attribute), 41
 FGM (nba_py.constants.StatCategory attribute), 42
 Finals (nba_py.constants.GameScope attribute), 37
 Finals (nba_py.constants.PlayoffRound attribute), 40
 FirstHalf (nba_py.constants.GameSegment attribute), 37
 FirstPick (nba_py.constants.DraftPick attribute), 36
 FirstQuarter (nba_py.constants.Period attribute), 39
 FirstRound (nba_py.constants.DraftPick attribute), 36
 Forward (nba_py.constants.PlayerPosition attribute), 39
 FourFactors (nba_py.constants.MeasureType attribute), 38
 FourthQuarter (nba_py.constants.Period attribute), 39
 FT_PCT (nba_py.constants.Sorter attribute), 41
 FT_PCT (nba_py.constants.StatCategory attribute), 42
 FTA (nba_py.constants.Sorter attribute), 41
 FTA (nba_py.constants.StatCategory attribute), 42
 FTM (nba_py.constants.Sorter attribute), 41
 FTM (nba_py.constants.StatCategory attribute), 42
- ## G
- game_header() (nba_py.Scoreboard method), 3
 game_info() (nba_py.game.BoxscoreSummary method), 26
 Game_Scope (class in nba_py.constants), 37
 game_summary() (nba_py.game.BoxscoreSummary method), 26
 GameID (class in nba_py.constants), 37
 gamenumber() (nba_py.player.PlayerLastNGamesSplits method), 11
 gamenumber() (nba_py.team.TeamLastNGamesSplits method), 29
 GameScope (class in nba_py.constants), 37
 GameSegment (class in nba_py.constants), 37
 general_shooting() (nba_py.player.PlayerShotTracking method), 21
 get() (nba_py.constants.ShotClockRange method), 41
 get_player() (in module nba_py.player), 23
 GroupQuantity (class in nba_py.constants), 37
 Guard (nba_py.constants.PlayerPosition attribute), 39
- ## H
- headline_stats() (nba_py.player.PlayerSummary method), 21
 Height (class in nba_py.constants), 37
 history() (nba_py.team.TeamDetails method), 28
 hof() (nba_py.team.TeamDetails method), 28
 Home (nba_py.constants.Location attribute), 38
 hustle_stats_available() (nba_py.game.HustleStats method), 26
 hustle_stats_player_box_score() (nba_py.game.HustleStats method), 26
 hustle_stats_team_box_score() (nba_py.game.HustleStats method), 26
 HustleStats (class in nba_py.game), 26
- ## I
- inactive_players() (nba_py.game.BoxscoreSummary method), 26
 info() (nba_py.game.PlayByPlay method), 26
 info() (nba_py.game.PlayerTracking method), 26
 info() (nba_py.player.PlayerGameLogs method), 8
 info() (nba_py.player.PlayerList method), 12
 info() (nba_py.player.PlayerSummary method), 21
 info() (nba_py.team.TeamGameLogs method), 28
 info() (nba_py.team.TeamList method), 29
 info() (nba_py.team.TeamSeasons method), 31
 info() (nba_py.team.TeamSummary method), 32
- ## J
- January (nba_py.constants.Month attribute), 38
 July (nba_py.constants.Month attribute), 38
 June (nba_py.constants.Month attribute), 38
- ## L
- Last10 (nba_py.constants.Game_Scope attribute), 37
 Last10 (nba_py.constants.GameScope attribute), 37
 last10() (nba_py.player.PlayerLastNGamesSplits method), 12
 last10() (nba_py.team.TeamLastNGamesSplits method), 29
 Last10Sec (nba_py.constants.ClutchTime attribute), 35
 last10sec_deficit_3point() (nba_py.player.PlayerClutchSplits method), 6
 last10sec_deficit_3point() (nba_py.team.TeamClutchSplits method), 27
 last15() (nba_py.player.PlayerLastNGamesSplits method), 12
 last15() (nba_py.team.TeamLastNGamesSplits method), 29
 Last1Min (nba_py.constants.ClutchTime attribute), 35
 last1min_deficit_5point() (nba_py.player.PlayerClutchSplits method), 7
 last1min_deficit_5point() (nba_py.team.TeamClutchSplits method), 27
 last1min_plusminus_5point() (nba_py.player.PlayerClutchSplits method), 7
 last1min_plusminus_5point() (nba_py.team.TeamClutchSplits method), 27

last20() (nba_py.player.PlayerLastNGamesSplits method), 12

last20() (nba_py.team.TeamLastNGamesSplits method), 29

Last2Min (nba_py.constants.ClutchTime attribute), 35

Last30Sec (nba_py.constants.ClutchTime attribute), 35

last30sec_deficit_3point() (nba_py.player.PlayerClutchSplits method), 7

last30sec_deficit_3point() (nba_py.team.TeamClutchSplits method), 27

last30sec_plusminus_5point() (nba_py.player.PlayerClutchSplits method), 7

last30sec_plusminus_5point() (nba_py.team.TeamClutchSplits method), 27

Last3Min (nba_py.constants.ClutchTime attribute), 35

last3min_deficit_5point() (nba_py.player.PlayerClutchSplits method), 7

last3min_deficit_5point() (nba_py.team.TeamClutchSplits method), 27

last3min_plusminus_5point() (nba_py.player.PlayerClutchSplits method), 7

last3min_plusminus_5point() (nba_py.team.TeamClutchSplits method), 27

Last4Min (nba_py.constants.ClutchTime attribute), 35

last5() (nba_py.player.PlayerLastNGamesSplits method), 12

last5() (nba_py.team.TeamLastNGamesSplits method), 29

Last5Min (nba_py.constants.ClutchTime attribute), 35

last5min_deficit_5point() (nba_py.player.PlayerClutchSplits method), 7

last5min_deficit_5point() (nba_py.team.TeamClutchSplits method), 28

last5min_plusminus_5point() (nba_py.player.PlayerClutchSplits method), 7

last5min_plusminus_5point() (nba_py.team.TeamClutchSplits method), 28

last_meeting() (nba_py.game.BoxscoreSummary method), 26

last_meeting() (nba_py.Scoreboard method), 3

LastNGames (class in nba_py.constants), 37

League (class in nba_py.constants), 37

line_score() (nba_py.game.BoxscoreSummary method), 26

line_score() (nba_py.Scoreboard method), 3

lineups() (nba_py.team.TeamLineups method), 29

Location (class in nba_py.constants), 37

location() (nba_py.player.PlayerGeneralSplits method), 9

location() (nba_py.team.TeamGeneralSplits method), 28

Loss (nba_py.constants.Outcome attribute), 38

Lottery (nba_py.constants.DraftPick attribute), 36

M

March (nba_py.constants.Month attribute), 38

May (nba_py.constants.Month attribute), 38

MeasureType (class in nba_py.constants), 38

MinutesPer (nba_py.constants.PerMode attribute), 39

Misc (nba_py.constants.MeasureType attribute), 38

Month (class in nba_py.constants), 38

month() (nba_py.player.PlayerGeneralSplits method), 9

monthly() (nba_py.team.TeamGeneralSplits method), 28

N

NBA (nba_py.constants.League attribute), 37

nba_py (module), 3

nba_py.constants (module), 35

nba_py.game (module), 25

nba_py.player (module), 5

nba_py.team (module), 27

next_game() (nba_py.player.PlayerProfile method), 15

Northwest (nba_py.constants.VsDivision attribute), 42

November (nba_py.constants.Month attribute), 38

num_contested_rebounding() (nba_py.player.PlayerReboundTracking method), 17

O

October (nba_py.constants.Month attribute), 38

off_court() (nba_py.team.TeamPlayerOnOffDetail method), 30

off_court() (nba_py.team.TeamPlayerOnOffSummary method), 31

officials() (nba_py.game.BoxscoreSummary method), 26

on_court() (nba_py.team.TeamPlayerOnOffDetail method), 30

on_court() (nba_py.team.TeamPlayerOnOffSummary method), 31

on_off_court() (nba_py.player.PlayerVsPlayer method), 22

on_off_court() (nba_py.team.TeamVsPlayer method), 32

Opponent (nba_py.constants.MeasureType attribute), 38

OpponentTeamID (class in nba_py.constants), 38

OREB (nba_py.constants.Sorter attribute), 41

OREB (nba_py.constants.StatCategory attribute), 42

other_stats() (nba_py.game.BoxscoreSummary method), 26

Outcome (class in nba_py.constants), 38

overall() (nba_py.player.PlayerVsPlayer method), 22

overall() (nba_py.team.TeamLineups method), 29

overall() (nba_py.team.TeamVsPlayer method), 32

Overtime (nba_py.constants.GameSegment attribute), 37

Overtime() (nba_py.constants.Period method), 39

P

- PaceAdjust (class in nba_py.constants), 38
- Pacific (nba_py.constants.VsDivision attribute), 42
- passes_made() (nba_py.player.PlayerPassTracking method), 14
- passes_made() (nba_py.team.TeamPassTracking method), 30
- passes_received() (nba_py.player.PlayerPassTracking method), 14
- passes_recieved() (nba_py.team.TeamPassTracking method), 30
- Per100Plays (nba_py.constants.PerMode attribute), 39
- Per100Possessions (nba_py.constants.PerMode attribute), 39
- Per36 (nba_py.constants.PerMode attribute), 39
- Per40 (nba_py.constants.PerMode attribute), 39
- Per48 (nba_py.constants.PerMode attribute), 39
- PerGame (nba_py.constants.PerMode attribute), 39
- Period (class in nba_py.constants), 39
- PerMinute (nba_py.constants.PerMode attribute), 39
- PerMode (class in nba_py.constants), 38
- PerPlay (nba_py.constants.PerMode attribute), 39
- PerPossession (nba_py.constants.PerMode attribute), 39
- PF (nba_py.constants.ContextMeasure attribute), 36
- PF (nba_py.constants.StatCategory attribute), 42
- Picks11Thru20 (nba_py.constants.DraftPick attribute), 36
- Picks21Thru30 (nba_py.constants.DraftPick attribute), 36
- PlayByPlay (class in nba_py.game), 26
- Player (nba_py.constants.Player_or_Team attribute), 40
- Player (nba_py.constants.PlayerOrTeam attribute), 39
- player_info() (nba_py.player.PlayerVsPlayer method), 22
- Player_or_Team (class in nba_py.constants), 40
- player_stats() (nba_py.game.Boxscore method), 25
- PlayerCareer (class in nba_py.player), 5
- PlayerClutchSplits (class in nba_py.player), 5
- PlayerDefenseTracking (class in nba_py.player), 7
- PlayerExperience (class in nba_py.constants), 39
- PlayerGameLogs (class in nba_py.player), 8
- PlayerGeneralSplits (class in nba_py.player), 8
- PlayerInGameSplits (class in nba_py.player), 9
- PlayerLastNGamesSplits (class in nba_py.player), 10
- PlayerList (class in nba_py.player), 12
- PlayerNotFoundException, 12
- PlayerOpponentSplits (class in nba_py.player), 12
- PlayerOrTeam (class in nba_py.constants), 39
- PlayerPassTracking (class in nba_py.player), 13
- PlayerPerformanceSplits (class in nba_py.player), 14
- PlayerPosition (class in nba_py.constants), 39
- PlayerProfile (class in nba_py.player), 15
- PlayerReboundLogTracking (class in nba_py.player), 15
- PlayerReboundTracking (class in nba_py.player), 16
- PlayerScope (class in nba_py.constants), 39
- PlayerShootingSplits (class in nba_py.player), 18
- PlayerShotLogTracking (class in nba_py.player), 19
- PlayerShotTracking (class in nba_py.player), 20
- PlayerSummary (class in nba_py.player), 21
- PlayerTracking (class in nba_py.game), 26
- PlayerVsPlayer (class in nba_py.player), 21
- PlayerYearOverYearSplits (class in nba_py.player), 22
- PlayoffRound (class in nba_py.constants), 40
- Playoffs (nba_py.constants.SeasonType attribute), 41
- PlusMinus (class in nba_py.constants), 40
- points_against() (nba_py.player.PlayerPerformanceSplits method), 15
- points_against() (nba_py.team.TeamPerformanceSplits method), 30
- points_scored() (nba_py.player.PlayerPerformanceSplits method), 15
- points_scored() (nba_py.team.TeamPerformanceSplits method), 30
- post_season_career_totals() (nba_py.player.PlayerCareer method), 5
- post_season_rankings() (nba_py.player.PlayerCareer method), 5
- post_season_totals() (nba_py.player.PlayerCareer method), 5
- PostAllStar (nba_py.constants.SeasonSegment attribute), 40
- pre_post_all_star() (nba_py.player.PlayerGeneralSplits method), 9
- pre_post_all_star() (nba_py.team.TeamGeneralSplits method), 28
- PreAllStar (nba_py.constants.SeasonSegment attribute), 40
- preseason_career_totals() (nba_py.player.PlayerCareer method), 5
- preseason_season_totals() (nba_py.player.PlayerCareer method), 5
- PtMeasureType (class in nba_py.constants), 40
- PTS (nba_py.constants.Sorter attribute), 41
- PTS (nba_py.constants.StatCategory attribute), 42
- PTS_2ND_CHANCE (nba_py.constants.ContextMeasure attribute), 36
- PTS_FB (nba_py.constants.ContextMeasure attribute), 36
- PTS_OFF_TOV (nba_py.constants.ContextMeasure attribute), 36

Q

- QuarterFinals (nba_py.constants.PlayoffRound attribute), 40

R

- RangeType (class in nba_py.constants), 40
- Rank (class in nba_py.constants), 40
- REB (nba_py.constants.Sorter attribute), 41
- REB (nba_py.constants.StatCategory attribute), 42
- rebound_distance_rebounding() (nba_py.player.PlayerReboundTracking

- method), 18
 - rebound_distance_rebounding() (nba_py.team.TeamReboundTracking method), 31
 - Regular (nba_py.constants.SeasonType attribute), 41
 - regular_season_career_totals() (nba_py.player.PlayerCareer method), 5
 - regular_season_rankings() (nba_py.player.PlayerCareer method), 5
 - regular_season_totals() (nba_py.player.PlayerCareer method), 5
 - retired() (nba_py.team.TeamDetails method), 28
 - Rookie (nba_py.constants.PlayerExperience attribute), 39
 - Rookies (nba_py.constants.PlayerScope attribute), 40
 - Rookies (nba_py.constants.Scope attribute), 40
 - RookieYear (class in nba_py.constants), 40
 - roster() (nba_py.team.TeamCommonRoster method), 28
- S**
- Scope (class in nba_py.constants), 40
 - score_differential() (nba_py.player.PlayerPerformanceSplits method), 15
 - score_differential() (nba_py.team.TeamPerformanceSplits method), 30
 - Scoreboard (class in nba_py), 3
 - Scoring (nba_py.constants.MeasureType attribute), 38
 - Season (nba_py.constants.GameScope attribute), 37
 - season_highs() (nba_py.player.PlayerProfile method), 15
 - season_ranks() (nba_py.team.TeamSummary method), 32
 - season_series() (nba_py.game.BoxscoreSummary method), 26
 - season_totals() (nba_py.team.TeamPlayers method), 31
 - SeasonSegment (class in nba_py.constants), 40
 - SeasonType (class in nba_py.constants), 40
 - SecondHalf (nba_py.constants.GameSegment attribute), 37
 - SecondQuarter (nba_py.constants.Period attribute), 39
 - SecondRound (nba_py.constants.DraftPick attribute), 36
 - SemiFinals (nba_py.constants.PlayoffRound attribute), 40
 - September (nba_py.constants.Month attribute), 38
 - series_standings() (nba_py.Scoreboard method), 3
 - shot_5ft() (nba_py.player.PlayerShootingSplits method), 19
 - shot_5ft() (nba_py.team.TeamShootingSplits method), 32
 - shot_8ft() (nba_py.player.PlayerShootingSplits method), 19
 - shot_8ft() (nba_py.team.TeamShootingSplits method), 32
 - shot_area_off_court() (nba_py.player.PlayerVsPlayer method), 22
 - shot_area_off_court() (nba_py.team.TeamVsPlayer method), 32
 - shot_area_on_court() (nba_py.player.PlayerVsPlayer method), 22
 - shot_area_on_court() (nba_py.team.TeamVsPlayer method), 32
 - shot_area_overall() (nba_py.player.PlayerVsPlayer method), 22
 - shot_area_overall() (nba_py.team.TeamVsPlayer method), 32
 - shot_areas() (nba_py.player.PlayerShootingSplits method), 19
 - shot_areas() (nba_py.team.TeamShootingSplits method), 32
 - shot_clock_shooting() (nba_py.player.PlayerShotTracking method), 21
 - shot_clock_shooting() (nba_py.team.TeamShotTracking method), 32
 - shot_distance_off_court() (nba_py.player.PlayerVsPlayer method), 22
 - shot_distance_off_court() (nba_py.team.TeamVsPlayer method), 32
 - shot_distance_on_court() (nba_py.player.PlayerVsPlayer method), 22
 - shot_distance_on_court() (nba_py.team.TeamVsPlayer method), 32
 - shot_distance_overall() (nba_py.player.PlayerVsPlayer method), 22
 - shot_distance_overall() (nba_py.team.TeamVsPlayer method), 32
 - shot_distance_rebounding() (nba_py.player.PlayerReboundTracking method), 18
 - shot_distance_rebounding() (nba_py.team.TeamReboundTracking method), 31
 - shot_type_rebounding() (nba_py.player.PlayerReboundTracking method), 18
 - shot_type_rebounding() (nba_py.team.TeamReboundTracking method), 31
 - shot_type_summary() (nba_py.team.TeamShootingSplits method), 32
 - shot_types_detail() (nba_py.player.PlayerShootingSplits method), 19
 - shot_types_summary() (nba_py.player.PlayerShootingSplits method), 19
 - ShotClockOff (nba_py.constants.ShotClockRange attribute), 41
 - ShotClockRange (class in nba_py.constants), 41
 - social_sites() (nba_py.team.TeamDetails method), 28
 - Sophomore (nba_py.constants.PlayerExperience attribute), 39
 - Sorter (class in nba_py.constants), 41
 - Southeast (nba_py.constants.VsDivision attribute), 42
 - Southwest (nba_py.constants.VsDivision attribute), 42
 - SpeedDistance (nba_py.constants.PtMeasureType attribute), 40
 - sql_players_advanced() (nba_py.game.BoxscoreAdvanced

- method), 25
 - sql_players_four_factors()
 - (nba_py.game.BoxscoreFourFactors method), 25
 - sql_players_misc()
 - (nba_py.game.BoxscoreMisc method), 25
 - sql_players_scoring()
 - (nba_py.game.BoxscoreScoring method), 26
 - sql_players_usage()
 - (nba_py.game.BoxscoreUsage method), 26
 - sql_team_advanced()
 - (nba_py.game.BoxscoreAdvanced method), 25
 - sql_team_four_factors()
 - (nba_py.game.BoxscoreFourFactors method), 25
 - sql_team_misc()
 - (nba_py.game.BoxscoreMisc method), 25
 - sql_team_scoring()
 - (nba_py.game.BoxscoreScoring method), 26
 - sql_team_usage()
 - (nba_py.game.BoxscoreUsage method), 26
 - StarterBench (class in nba_py.constants), 41
 - Starters (nba_py.constants.StarterBench attribute), 41
 - starting_position()
 - (nba_py.player.PlayerGeneralSplits method), 9
 - StartPeriod (class in nba_py.constants), 41
 - StartRange (class in nba_py.constants), 41
 - StatCategory (class in nba_py.constants), 41
 - STL (nba_py.constants.Sorter attribute), 41
 - STL (nba_py.constants.StatCategory attribute), 42
 - STL_TOV (nba_py.constants.StatCategory attribute), 42
- T**
- Team (nba_py.constants.Player_or_Team attribute), 40
 - Team (nba_py.constants.PlayerOrTeam attribute), 39
 - team_starter_bench_stats()
 - (nba_py.game.Boxscore method), 25
 - team_stats()
 - (nba_py.game.Boxscore method), 25
 - TeamClutchSplits (class in nba_py.team), 27
 - TeamCommonRoster (class in nba_py.team), 28
 - TeamDetails (class in nba_py.team), 28
 - TeamGameLogs (class in nba_py.team), 28
 - TeamGeneralSplits (class in nba_py.team), 28
 - TeamID (class in nba_py.constants), 42
 - TeamInGameSplits (class in nba_py.team), 28
 - TeamLastNGamesSplits (class in nba_py.team), 29
 - TeamLineups (class in nba_py.team), 29
 - TeamList (class in nba_py.team), 29
 - TeamOpponentSplits (class in nba_py.team), 29
 - TeamPassTracking (class in nba_py.team), 30
 - TeamPerformanceSplits (class in nba_py.team), 30
 - TeamPlayerOnOffDetail (class in nba_py.team), 30
 - TeamPlayerOnOffSummary (class in nba_py.team), 30
 - TeamPlayers (class in nba_py.team), 31
 - TeamReboundTracking (class in nba_py.team), 31
 - TeamSeasons (class in nba_py.team), 31
 - TeamShootingSplits (class in nba_py.team), 31
 - TeamShotTracking (class in nba_py.team), 32
 - TeamSummary (class in nba_py.team), 32
 - TeamVsPlayer (class in nba_py.team), 32
 - TeamYearOverYearSplits (class in nba_py.team), 32
 - ThirdQuarter (nba_py.constants.Period attribute), 39
 - Top10 (nba_py.constants.DraftPick attribute), 36
 - Top15 (nba_py.constants.DraftPick attribute), 36
 - Top20 (nba_py.constants.DraftPick attribute), 36
 - Top25 (nba_py.constants.DraftPick attribute), 36
 - Top5 (nba_py.constants.DraftPick attribute), 36
 - Totals (nba_py.constants.PerMode attribute), 39
 - touch_time_shooting()
 - (nba_py.player.PlayerShotTracking method), 21
 - touch_time_shooting()
 - (nba_py.team.TeamShotTracking method), 32
 - TOV (nba_py.constants.Sorter attribute), 41
 - TOV (nba_py.constants.StatCategory attribute), 42
 - TS_PCT (nba_py.constants.ContextMeasure attribute), 36
- U**
- Undrafted (nba_py.constants.DraftPick attribute), 37
 - Usage (nba_py.constants.MeasureType attribute), 38
- V**
- Veteran (nba_py.constants.PlayerExperience attribute), 39
 - vs_player_info()
 - (nba_py.player.PlayerVsPlayer method), 22
 - vs_player_overall()
 - (nba_py.team.TeamVsPlayer method), 32
 - VsConference (class in nba_py.constants), 42
 - VsDivision (class in nba_py.constants), 42
- W**
- Weight (class in nba_py.constants), 43
 - West (nba_py.constants.VsConference attribute), 42
 - west_conf_standings_by_day()
 - (nba_py.Scoreboard method), 3
 - Win (nba_py.constants.Outcome attribute), 38
 - win_losses()
 - (nba_py.player.PlayerGeneralSplits method), 9
 - wins_losses()
 - (nba_py.team.TeamGeneralSplits method), 28
- Y**
- Yesterday (nba_py.constants.Game_Scope attribute), 37
 - Yesterday (nba_py.constants.GameScope attribute), 37