BioArch

Release 0.0.35

May 02, 2020

Contents:

CHAPTER 1

bioarch

1.1 bioarch package

1.1.1 Submodules

1.1.2 bioarch.age module

class bioarch.age.**AgeCategory**
Bases: enum.Enum

An enumeration.

ADULT = 6

MIDDLE = 3

MIDDLE_OLD = 4

OLD = 5

UNKNOWN = 0

YOUNG = 1

YOUNG_ADULT = 2

as_quad()

dtype = <function AgeCategory.dtype>

parse = <function AgeCategory.parse>

class bioarch.age.**EstimatedAge**(*category: str, ranged: Optional[str]*)
Bases: object

docstring for EstimatedAge

MAX_AGE = 100

static empty()

to_pd_data_frame(*index*)

1.1.3 bioarch.individual module

class bioarch.individual.**AgeSexStature**(*osteological_sex: bioarch.individual.OsteologicalSex, age: bioarch.age.EstimatedAge, femur: bioarch.individual.LongBoneMeasurement, humerus: bioarch.individual.LongBoneMeasurement, tibia: bioarch.individual.LongBoneMeasurement, stature: Optional[str], body_mass: Optional[str]*)

Bases: object

docstring for AgeSexStature

age

body_mass

static empty()

femur

humerus

osteological_sex

stature

tibia

to_pd_data_frame(*index*)

class bioarch.individual.**BurialInfo**(*site_name: str*, *site_id: str*)

Bases: object

docstring for BurialInfo

to_pd_series(*prefix=''*)

class bioarch.individual.**Individual**(*_id: str*, *site: bioarch.individual.BurialInfo*, *age_sex_stature: bioarch.individual.AgeSexStature*, *mouth: bioarch.mouth.Mouth*, *occupational_markers: bioarch.occupational_markers.OccupationalMarkers*, *joints: bioarch.joints.Joints*, *trauma: bioarch.trauma.Trauma*, *context: bioarch.context.Context*)

Bases: object

docstring for Individual

to_pd_data_frame()

class bioarch.individual.**LongBoneMeasurement**(*_max: Optional[float], bi: Optional[float], head: Optional[float], distal: Optional[float]*)

Bases: object

docstring for LongBoneMeasurement

static empty()

static empty_lr()

to_pd_series(*prefix=''*)

class bioarch.individual.**OsteologicalSex**(*pelvic: Optional[bioarch.sex.Sex], cranium: Optional[bioarch.sex.Sex], combined: Optional[bioarch.sex.Sex]*)

Bases: object

docstring for OsteologicalSex

static empty()

to_pd_data_frame(*index*)

1.1.4 bioarch.left_right module

class bioarch.left_right.**LeftRight**(*left: Optional[T], right: Optional[T]*)

Bases: typing.Generic

Wrapper around measurements taken on both the left and right sides. This augments the two measurements by adding a meta-measurement with the "avg"/"best" combination of both.

avg() → Optional[T]

meta-measurement with the "avg"/"best" combination of both.

left

right

bioarch.left_right.**best_effort_avg**(*left: Optional[T], right: Optional[T]*) → Optional[T]

1.1.5 bioarch.mouth module

class bioarch.mouth.**Mouth**(*teeth: List[bioarch.mouth.Tooth]*)

Bases: object

Object to hold the 32 teeth

static empty()

to_pd_series(*prefix=''*)

class bioarch.mouth.**Tooth**

Bases: object

docstring for Tooth

abcess

'NA', '0', '1'

calculus

'NA', '0', '1', '2', '3'

cavities

'NA', '0', '1'

eh

'NA', '0', '1'

static empty()

to_pd_series(*prefix=''*)

tooth
'NA', '0', '1', 'A', 'B1', 'B2', 'C', 'D', 'E', 'F', 'G', 'H', 'I'

1.1.6 bioarch.occupational_markers module

class bioarch.occupational_markers.**EnthesialMarker**(*value: Union[int, float], is_s: bool = False, is_oe: bool = False*)

Bases: object

docstring for EnthesialMarker

as_num() → float

static avg(*left: Optional[EnthesialMarker], right: Optional[EnthesialMarker]*) → Optional[bioarch.occupational_markers.EnthesialMarker]

static parse(*value: Any*) → Optional[bioarch.occupational_markers.EnthesialMarker]

class bioarch.occupational_markers.**OccupationalMarkers**(*c_trapezius: bioarch.left_right.LeftRight[bioarch.occupational_m*
c_o_deltiod: bioarch.left_right.LeftRight[bioarch.occupational_m
c_o_pectoralis_major: bioarch.left_right.LeftRight[bioarch.occupational_m
c_costoclaviclar_lig: bioarch.left_right.LeftRight[bioarch.occupational_m
c_subcalvius: bioarch.left_right.LeftRight[bioarch.occupational_m
c_conoid_lig: bioarch.left_right.LeftRight[bioarch.occupational_m
c_trapezoid_lig: bioarch.left_right.LeftRight[bioarch.occupational_m
s_pectoralis_minor: bioarch.left_right.LeftRight[bioarch.occupational_m
s_levator_scapulae: bioarch.left_right.LeftRight[bioarch.occupational_m
s_triceps_long_head: bioarch.left_right.LeftRight[bioarch.occupational_m
s_trapezius: bioarch.left_right.LeftRight[bioarch.occupational_m
h_subscapularis: bioarch.left_right.LeftRight[bioarch.occupational_m
h_teres_major: bioarch.left_right.LeftRight[bioarch.occupational_m
h_latissimus_dorsi: bioarch.left_right.LeftRight[bioarch.occupational_m
h_pectoralis_major: bioarch.left_right.LeftRight[bioarch.occupational_m
h_deltoid: bioarch.left_right.LeftRight[bioarch.occupational_m
h_coracobrachialis: bioarch.left_right.LeftRight[bioarch.occupational_m
h_supraspinatus: bioarch.left_right.LeftRight[bioarch.occupational_m
h_infraspinatus: bioarch.left_right.LeftRight[bioarch.occupational_m
h_teres_minor: bioarch.left_right.LeftRight[bioarch.occupational_m
h_o_extensor: bioarch.left_right.LeftRight[bioarch.occupational_m
h_o_flexor: bioarch.left_right.LeftRight[bioarch.occupational_m
u_brachialis: bioarch.left_right.LeftRight[bioarch.occupational_m
u_o_pronator_quadrataus: bioarch.left_right.LeftRight[bioarch.occupational_m
u_triceps_brachii: bioarch.left_right.LeftRight[bioarch.occupational_m
u_anconeus: bioarch.left_right.LeftRight[bioarch.occupational_m
u_o_supinator: bioarch.left_right.LeftRight[bioarch.occupational_m
r_biceps_brachii: bioarch.left_right.LeftRight[bioarch.occupational_m
r_supinator: bioarch.left_right.LeftRight[bioarch.occupational_m
r_pronator_teres: bioarch.left_right.LeftRight[bioarch.occupational_m

Bases: object

docstring for OccupationalMarkers

static empty() → bioarch.occupational_markers.OccupationalMarkers

to_pd_data_frame(*index*) → pandas.core.series.Series

1.1.7 bioarch.sex module

class bioarch.sex.**Sex**

Bases: enum.Enum

An enumeration.

FEMALE = 0

FEMALE_ASSUMED = 20

FEMALE_LIKELY = 10

MALE = 100

MALE_ASSUMED = 80

MALE_LIKELY = 90

UNKNOWN = 50

as_bin()

dtype = <function Sex.dtype>

parse = <function Sex.parse>

1.1.8 Module contents

class bioarch.**AgeCategory**

Bases: enum.Enum

An enumeration.

ADULT = 6

MIDDLE = 3

MIDDLE_OLD = 4

OLD = 5

UNKNOWN = 0

YOUNG = 1

YOUNG_ADULT = 2

as_quad()

dtype = <function AgeCategory.dtype>

parse = <function AgeCategory.parse>

class bioarch.**EstimatedAge**(*category: str, ranged: Optional[str]*)

Bases: object

docstring for EstimatedAge

MAX_AGE = 100

static empty()

to_pd_data_frame(*index*)

class bioarch.**BodyPosition**

Bases: enum.Enum

An enumeration.

CROUCHED = 2

CROUCHED_LEFT_SIDE = 3

CROUCHED_RIGHT_SIDE = 4

STOMACH = 5

SUPINE = 0

SUPINE_FLEXED_LEGS = 1

dtype = <function BodyPosition.dtype>

parse = <function BodyPosition.parse>

class bioarch.**CompassBearing**

Bases: enum.Enum

An enumeration.

EAST = 2

NORTH = 0

NORTH_EAST = 1

NORTH_WEST = 7

SOUTH = 4

SOUTH_EAST = 3

SOUTH_WEST = 5

WEST = 6

dtype = <function CompassBearing.dtype>

parse = <function CompassBearing.parse>

to_short_code()

class bioarch.**Context**(*body_position: Optional[bioarch.context.BodyPosition], body_orientation: Optional[bioarch.context.CompassBearing], disturbed: Optional[bioarch.context.Present], decapitation: Optional[bioarch.context.Present], double_grave: Optional[bioarch.context.Present], stone_layer: Optional[bioarch.context.Present], grave_goods: Dict[str, Optional[Any]]*)

Bases: object

docstring for Context

static empty()

static group(*value*)

to_pd_data_frame(*index*)

class bioarch.**Present**
Bases: enum.Enum

An enumeration.

NOT_PRESENT = 0

PRESENT = 1

dtype = <function Present.dtype>

parse = <function Present.parse>

class bioarch.**AgeSexStature**(*osteological_sex: bioarch.individual.OsteologicalSex, age: bioarch.age.EstimatedAge, femur: bioarch.individual.LongBoneMeasurement, humerus: bioarch.individual.LongBoneMeasurement, tibia: bioarch.individual.LongBoneMeasurement, stature: Optional[str], body_mass: Optional[str]*)
Bases: object

docstring for AgeSexStature

age

body_mass

static empty()

femur

humerus

osteological_sex

stature

tibia

to_pd_data_frame(*index*)

class bioarch.**BurialInfo**(*site_name: str*, *site_id: str*)
Bases: object

docstring for BurialInfo

to_pd_series(*prefix=''*)

class bioarch.**Individual**(*_id: str*, *site: bioarch.individual.BurialInfo*, *age_sex_stature: bioarch.individual.AgeSexStature*, *mouth: bioarch.mouth.Mouth*, *occupational_markers: bioarch.occupational_markers.OccupationalMarkers*, *joints: bioarch.joints.Joints*, *trauma: bioarch.trauma.Trauma*, *context: bioarch.context.Context*)
Bases: object

docstring for Individual

to_pd_data_frame()

class bioarch.**LongBoneMeasurement**(*_max: Optional[float], bi: Optional[float], head: Optional[float], distal: Optional[float]*)
Bases: object

docstring for LongBoneMeasurement

static empty()

static empty_lr()

to_pd_series(*prefix=''*)

class bioarch.**OsteologicalSex**(*pelvic: Optional[bioarch.sex.Sex], cranium: Optional[bioarch.sex.Sex], combined: Optional[bioarch.sex.Sex]*)

Bases: object

docstring for OsteologicalSex

static empty()

to_pd_data_frame(*index*)

class bioarch.**JointCondition**

Bases: enum.Enum

1 - slight/mild 2 - medium 3 - extreme 4 - fused 5 - SCHMORL NODES 6 - FRACTURE

EXTREME = 3

FRACTURE = 6

FUSED = 4

MEDIUM = 2

MILD = 1

NORMAL = 0

SCHMORL_NODES = 5

avg = <function JointCondition.avg>

dtype = <function JointCondition.dtype>

parse = <function JointCondition.parse>

class bioarch.**Joints**(*shoulder: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], elbow: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], wrist: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], hip: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], knee: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], ankle: bioarch.left_right.LeftRight[bioarch.joints.JointCondition][bioarch.joints.JointCondition], sacro_illiac: bioarch.joints.JointCondition, c1_3: bioarch.joints.JointCondition, c4_7: bioarch.joints.JointCondition, t1_4: bioarch.joints.JointCondition, t5_8: bioarch.joints.JointCondition, t9_12: bioarch.joints.JointCondition, l1_5: bioarch.joints.JointCondition*)

Bases: object

docstring for Joints

static empty()

to_pd_data_frame(*index*)

class bioarch.**EnthesialMarker**(*value: Union[int, float], is_s: bool = False, is_oe: bool = False*)

Bases: object

docstring for EnthesialMarker

as_num() → float

static avg(*left: Optional[EnthesialMarker], right: Optional[EnthesialMarker]*) → Optional[bioarch.occupational_markers.EnthesialMarker]

static parse(*value: Any*) → Optional[bioarch.occupational_markers.EnthesialMarker]

class bioarch.**OccupationalMarkers**(*c_trapezius: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM*
c_o_deltiod: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
c_o_pectoralis_major: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
c_costoclaviclar_lig: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
c_subcalvius: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
c_conoid_lig: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesiall
c_trapezoid_lig: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesi
s_pectoralis_minor: bioarch.left_right.LeftRight[bioarch.occupational_markers.Ent
s_levator_scapulae: bioarch.left_right.LeftRight[bioarch.occupational_markers.Ent
s_triceps_long_head: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
s_trapezius: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
h_subscapularis: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
h_teres_major: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesia
h_latissimus_dorsi: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enth
h_pectoralis_major: bioarch.left_right.LeftRight[bioarch.occupational_markers.Ent
h_deltoid: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMar
h_coracobrachialis: bioarch.left_right.LeftRight[bioarch.occupational_markers.Ent
h_supraspinatus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
h_infraspinatus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesi
h_teres_minor: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesia
h_o_extensor: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialI
h_o_flexor: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
u_brachialis: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
u_o_pronator_quadrataus: bioarch.left_right.LeftRight[bioarch.occupational_marke
u_triceps_brachii: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthe
u_anconeus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
u_o_supinator: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthesia
r_biceps_brachii: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
r_supinator: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
r_pronator_teres: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
r_pronator_quadratus: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
r_brachoradialis: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
f_gluteus_minimus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enth
f_gluteus_medius: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthe
f_piriformus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialM
f_obturator_internus: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
f_obturator_externus: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
f_quadratis_femoris: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
f_ilioposas: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
f_gluteus_maximus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enth
f_pectineus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
f_o_vastus_medialis: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
f_o_vastus_lateralis: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
f_adductor_magnus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Ent
f_o_gastrocnemius: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enth
f_o_plantaris: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialI
f_o_popliteus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialI
t_tensor_fascia_latae: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
t_quadriceps: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialI
t_sartorius: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
t_gracilis: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMar
t_semitendinosus: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthe
t_o_tibialus_anterior: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
t_biceps_femoris: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes
t_semimembranosus: bioarch.left_right.LeftRight[bioarch.occupational_markers.En
t_popliteus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
t_o_soleus: bioarch.left_right.LeftRight[bioarch.occupational_markers.EnthesialMa
t_o_tibialis_posterior: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
t_o_flexor_digitorium: bioarch.left_right.LeftRight[bioarch.occupational_markers.E
f_biceps_femoris: bioarch.left_right.LeftRight[bioarch.occupational_markers.Enthes

Bases: object

docstring for OccupationalMarkers

static empty() → bioarch.occupational_markers.OccupationalMarkers

to_pd_data_frame(*index*) → pandas.core.series.Series

class bioarch.**LeftRight**(*left: Optional[T], right: Optional[T]*)
Bases: typing.Generic

Wrapper around measurements taken on both the left and right sides. This augments the two measurements by adding a meta-measurement with the "avg"/"best" combination of both.

avg() → Optional[T]
meta-measurement with the "avg"/"best" combination of both.

left

right

class bioarch.**Mouth**(*teeth: List[bioarch.mouth.Tooth]*)
Bases: object

Object to hold the 32 teeth

static empty()

to_pd_series(*prefix=''*)

class bioarch.**Tooth**
Bases: object

docstring for Tooth

abcess
'NA', '0', '1'

calculus
'NA', '0', '1', '2', '3'

cavities
'NA', '0', '1'

eh
'NA', '0', '1'

static empty()

to_pd_series(*prefix=''*)

tooth
'NA', '0', '1', 'A', 'B1', 'B2', 'C', 'D', 'E', 'F', 'G', 'H', 'I'

class bioarch.**Sex**
Bases: enum.Enum

An enumeration.

FEMALE = 0

FEMALE_ASSUMED = 20

FEMALE_LIKELY = 10

MALE = 100

MALE_ASSUMED = 80

MALE_LIKELY = 90

UNKNOWN = 50

as_bin()

dtype = <function Sex.dtype>

parse = <function Sex.parse>

class bioarch.**Trauma**(*facial_bones: bioarch.trauma.TraumaCategory, clavicle: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory], scapula: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategor humerus: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCateg ulna: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory], radius: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory femur: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory tibia: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory], fibula: bioarch.left_right.LeftRight[bioarch.trauma.TraumaCategory][bioarch.trauma.TraumaCategory ribs: bioarch.trauma.TraumaCategory, vertabrae: bioarch.trauma.TraumaCategory*)

Bases: object

docstring for Trauma

static empty()

to_pd_data_frame(*index*)

class bioarch.**TraumaCategory**

Bases: enum.Enum

An enumeration.

BLUNT_FORCE_TRAUMA = 6

BONY_GROWTH = 10

CRIBA = 5

FRACTURE = 3

FUSED = 11

INFECTION = 2

NORMAL = 1

NOT_PRESENT = -1

OSTEOCHONDRITIS_DESSICANS = 12

PARTIAL_BONE = 0.5

SHARP_FORCE_TRAUMA = 7

TREPONATION = 8

UNFUSED = 9

UNHEALED_FRACTURE = 4

avg = <function TraumaCategory.avg>

dtype = <function TraumaCategory.dtype>

parse = <function TraumaCategory.parse>

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

b

Index

P

R

S

T

U

W

Y