

# Package ‘hetu’

December 4, 2024

**Type** Package

**Title** Structural Handling of Finnish Personal Identity Codes

**Version** 1.1.0

**Date** 2024-12-03

**MailingList** rOpenGov <ropengov-forum@googlegroups.com>

**Description** Structural handling of Finnish identity codes (natural persons and organizations); extract information, check ID validity and diagnostics.

**License** BSD\_2\_clause + file LICENSE

**VignetteBuilder** knitr

**Encoding** UTF-8

**BugReports** <https://github.com/ropengov/hetu/issues>

**URL** <https://ropengov.github.io/hetu/>, <https://github.com/ropengov/hetu>

**Depends** R (>= 3.6.0)

**Imports** lubridate, checkmate, parallel, methods

**Suggests** Cairo, knitr, testthat, rmarkdown, covr, dplyr

**RoxygenNote** 7.3.2

**X-schema.org-isPartOf** <http://ropengov.org/>

**X-schema.org-keywords** ropengov

**Config/Needs/website** magick, ropengov/rogtemplate

**NeedsCompilation** no

**Author** Pyry Kantanen [aut, cre] (<<https://orcid.org/0000-0003-2853-2765>>),  
Mans Magnusson [aut],  
Jussi Paananen [aut],  
Juho Kopra [ctb],  
Oskari Luomala [ctb],  
Tuomo Nieminen [ctb],  
Leo Lahti [aut] (<<https://orcid.org/0000-0001-5537-637X>>)

**Maintainer** Pyry Kantanen <pyry.kantanen@gmail.com>

**Repository** CRAN

**Date/Publication** 2024-12-04 08:20:10 UTC

## Contents

bid_ctrl . . . . .	2
hetu . . . . .	3
hetu_control_char . . . . .	5
hetu_diagnostic . . . . .	6
is.diagnostic . . . . .	7
pin_age . . . . .	8
pin_ctrl . . . . .	9
pin_date . . . . .	10
pin_sex . . . . .	10
plot.diagnostic . . . . .	11
rbid . . . . .	12
rpin . . . . .	12
sat_u_control_char . . . . .	14
sat_u_ctrl . . . . .	15
<b>Index</b>	<b>16</b>

---

bid_ctrl	<i>Check Validity of Finnish Business ID (Y-tunnus)</i>
----------	---

---

### Description

A function that checks whether a bid (Finnish Business ID) is valid. Returns TRUE or FALSE.

### Usage

```
bid_ctrl(bid)
```

### Arguments

bid                    a vector of 1 or more business identity numbers

### Examples

```
bid_ctrl(c("0000000-0", "0000001-9")) # TRUE TRUE
bid_ctrl("0737546-1") # FALSE
```

**Description**

Extract embedded information from Finnish personal identity codes (hetu).

**Usage**

```
hetu(  
  pin,  
  extract = NULL,  
  allow.temp = FALSE,  
  diagnostic = FALSE,  
  as.factor = FALSE  
)
```

**Arguments**

pin	Finnish personal identity code(s) as a character vector
extract	Extract only selected part of the information. Valid values are "hetu", "sex", "p.num", "ctrl.char", "date", "day", "month", "year", "century", "is.temp". If NULL (default), returns all information.
allow.temp	Allow artificial or temporary PINs (personal numbers 900-999). If FALSE (default), only PINs intended for official use (personal numbers 002-899) are allowed.
diagnostic	Print additional information about possible problems in PINs. The checks are "valid.p.num", "valid.ctrl.char", "correct.ctrl.char", "valid.date", "valid.day", "valid.month", "valid.length", "valid.century". Default is FALSE which returns no diagnostic information.
as.factor	Makes fields "sex", "p.num", "ctrl.char" and "century" into factors for slightly reduced memory footprint. Default is FALSE.

**Details**

Starting from 1st of January 2023, an amendment to the government decree on the Population Information System (128/2010) has expanded the number of available century markers (See references: Valtioneuvoston asetus VM/2022/124) and scrapped some old practices.

For the users of this package the most visible change will be that people born in the 1900s can now be assigned with "Y", "X", "W", "V" or "U", in addition to the old "/" (slash) marker. People born in the 2000s can be assigned with "B", "C", "D", "E" or "F", in addition to the old marker, "A". For people born in the 1800s "+" (plus sign) remains the only valid marker. The amendment does not affect already existing personal identity codes.

The change was done to mitigate for the diminishing pool of available, unique identity codes. For historical reasons, the century marker of the code was not always taken into account when determining the uniqueness of the number. This meant that individual number parts were not recycled

between people born in different centuries, diminishing the amount of available numbers for people born in the new century. For example, if a female born in the 1st of January 1901 was assigned with the personal identity code "010101-0101" (individual code part "010"), a female born in 1st of January 2001 could not be assigned with the code "010101A0101" because it would contain the same individual code as the person born in 1901 and individual codes could not be recycled. With the amended decree the uniqueness of the personal identity code is considered by looking at the personal identity code as a whole. This means that from now on it would be permissible to have personal identity codes such as "100190-999P" and "100190Y999P" at the same time, denoting two different individuals (see references: Digital and population data services agency announcement).

In practice, codes with new separators will be issued only when the ranges with currently used separators run out. This means that it might take a while until we see people born in the 2000s assigned with the century marker "C" or people born in the 1900s assigned with the century marker "X", as there are still plenty of numbers in ranges "B" and "Y" as well, in addition to some numbers being left in the original ranges of "A" and "-". The first personal identity code with a new separator "Y" was assigned in December 2023 (see Digi- ja väestötietovirasto 2023).

The result of all this is that the hetu package may now give "unrealistic" personal identity codes in the sense that some codes are not yet actually in use. However, it is not the aim of this package to simulate the actual distributions of personal identity codes and their century markers in the population (the actually used and unused codes are unknown to us), but to provide a tool that can be used to extract data from these codes, should the user encounter them at some point. Writing further sanity checks is probably a good idea for people who are interested in detecting unusual patterns in their databases and registries.

## Value

Finnish personal identity code data.frame, or if extract parameter is set, the requested part of the information as a vector. Returns an error or NA if the given character vector is not a valid Finnish personal identity code.

hetu	Finnish personal identity code as a character vector. A correct pin should be in the form DDMMYYCZZZQ, where DDMMYY stands for date, C for century sign, ZZZ for personal number and Q for control character.
sex	sex of the person as a character vector ("Male" or "Female")
p.num	Personal number (individual number) part of the identity code
ctrl.char	Control character for the personal identity code
date	Birthdate
day	Day of the birthdate
month	Month of the birthdate
year	Year of the birthdate
century	Century character determining the century (1800s, 1900s or 2000s) of the person's birth. See details for more information
valid.pin	Does the personal identity code pass all validity checks: (TRUE or FALSE)

## Author(s)

Pyry Kantanen, Jussi Paananen

## References

Valtioneuvoston asetus VM/2022/124 [Valtioneuvoston asetus VM/2022/124](#)

Digi- ja väestötietovirasto. (2023). [Uudet välimerkit takaavat henkilötunnusten riittävyyden - ensimmäinen uudenlainen henkilötunnus myönnettiin tällä viikolla](#)

Digital and Population Data Services Agency. [Reform of the separators in the personal identity code](#)

## See Also

[pin\\_ctrl](#) Validating Finnish personal identity codes. [rhetu](#) Generating random Finnish personal identity codes.

## Examples

```
hetu("111111-111C")
hetu("111111-111C")$date
hetu("111111-111C")$sex
# Same as previous, but using extract argument
hetu("111111-111C", extract="sex")
# Process a vector of hetu's
hetu(c("010101-0101", "111111-111C"))
# Process a vector of hetu's and extract sex information from each
hetu(c("010101-0101", "111111-111C"), extract="sex")
# Process codes with new century markers
new_codes <- c("010594Y9032", "010594Y9021", "020594X903P")
hetu(new_codes)
```

---

hetu\_control\_char      *Calculate Control Character for Personal Identity Code*

---

## Description

Calculate a valid control character for an incomplete Finnish personal identity codes (hetu).

## Usage

```
hetu_control_char(pin, with.century = TRUE)
```

## Arguments

pin	An incomplete PIN that ONLY has a date, century marker (optional, see parameter with.century) and personal number
with.century	If TRUE (default), the function assumes that the PIN input contains a century marker (DDMMYYQZZZ). If FALSE, the function assumes that the PIN contains only date and personal number (DDMMYYZZZ).

**Details**

This method of calculating the control character was devised by mathematician Erkki Pale (1962) to detect input errors but also to detect errors produced by early punch card machines. The long number produced by writing the birth date and the personal number together are divided by 31 and the remainder is used to look up the control character from a separate table containing alphanumeric characters except letters G, I, O, Q and Z.

The method of calculating the control character does not need century character and therefore the function has an option to omit it.

**Value**

Control character, either a number 0-9 or a letter.

**Author(s)**

Pyry Kantanen

**See Also**

[hetu](#) For extracting information from Finnish personal identity codes.

**Examples**

```
hetu_control_char("010101-010")
hetu_control_char("010101010", with.century = FALSE)
```

---

hetu\_diagnostic

*Diagnostics Tool for Personal Identity Codes*

---

**Description**

Prints information on the tests that are used to confirm or reject the validity of each personal identity code.

**Usage**

```
hetu_diagnostic(pin, extract = NULL)
```

```
pin_diagnostic(pin, extract = NULL)
```

**Arguments**

pin	Finnish personal identification number as a character vector, or vector of identification numbers as a character vectors
extract	Extract only selected part of the diagnostic information. Valid values are "hetu", "is.temp", "valid.p.num", "valid.ctrl.char", "correct.ctrl.char", "valid.date", "valid.day", "valid.month", "valid.length", "valid.century". If NULL (default), returns all information.

**Value**

A data.frame containing diagnostic checks about PINs.

**See Also**

[hetu](#) for the main function on which hetu\_diagnostic relies on.

**Examples**

```
diagnosis_example <- c("010101-0102", "111111-111Q",
  "010101B0101", "320101-0101", "011301-0101",
  "010101-01010", "010101-0011", "010101-9011", "010101-901S")
## Print all diagnostics for various fake personal identity codes
hetu_diagnostic(diagnosis_example)
# Extract century-related checks
hetu_diagnostic(diagnosis_example, extract = "valid.century")
# Print a summary in natural language
summary(hetu_diagnostic(diagnosis_example))
diagnosis_example <- c("010101-0102", "111111-111Q",
  "010101B0101", "320101-0101", "011301-0101",
  "010101-01010", "010101-0011")
## Print all diagnoses
pin_diagnostic(diagnosis_example)
```

---

is.diagnostic

*Is an Object from Class "diagnostic"?*

---

**Description**

Returns TRUE if the object has class "diagnostic"

**Usage**

```
is.diagnostic(object)
```

**Arguments**

object            Object to be tested

**Value**

TRUE or FALSE

---

`pin_age`*Extract Age from Personal Identity Code*

---

**Description**

Calculate age in years, months, weeks or days from personal identity codes.

**Usage**

```
pin_age(pin, date = Sys.Date(), timespan = "years", allow.temp = FALSE)
```

```
hetu_age(pin, date = Sys.Date(), timespan = "years", allow.temp = FALSE)
```

**Arguments**

<code>pin</code>	Finnish personal identity code(s) as a character vector
<code>date</code>	Date at which age is calculated. If a vector is provided it must be of the same length as the <code>pin</code> argument.
<code>timespan</code>	Timespan to use to calculate age. The possible timespans are: <ul style="list-style-type: none"><li>• years (Default)</li><li>• months</li><li>• weeks</li><li>• days</li></ul>
<code>allow.temp</code>	Allow artificial or temporary PINs (personal numbers 900-999). If FALSE (default), only PINs intended for official use (personal numbers 002-899) are allowed.

**Value**

Age as an integer vector.

**Examples**

```
ex_pin <- c("010101-0101", "111111-111C")
pin_age(ex_pin, date = "2012-01-01")
```

```
ex_pin <- c("010101-0101", "111111-111C")
hetu_age(ex_pin, date = "2012-01-01")
```



---

`pin_ctrl`*Check Validity of Personal Identity Code*

---

**Description**

Validate Finnish personal identity codes (hetu).

**Usage**

```
pin_ctrl(pin, allow.temp = FALSE)
```

```
hetu_ctrl(pin, allow.temp = FALSE)
```

**Arguments**

<code>pin</code>	Finnish personal identity code(s) as a character vector
<code>allow.temp</code>	If TRUE, temporary PINs (personal numbers 900-999) are handled similarly to regular PINs (personal numbers 002-899), meaning that otherwise valid temporary PIN will return a TRUE. Default is FALSE.

**Value**

A logical vector indicating whether the input vector contains valid Finnish personal identity codes.

**Author(s)**

Pyry Kantanen

**See Also**

[hetu](#) For extracting information from Finnish personal identity codes.

**Examples**

```
pin_ctrl("010101-0101") # TRUE
pin_ctrl("010101-010A") # FALSE
pin_ctrl(c("010101-0101", "010101-010A")) # TRUE FALSE
hetu_ctrl("010101-0101") # TRUE
hetu_ctrl("010101-010A") # FALSE
hetu_ctrl(c("010101-0101", "010101-010A")) # TRUE FALSE
```

---

pin_date	<i>Extract Date of Birth from Personal Identity Code</i>
----------	--

---

**Description**

Returns the date of birth in date format.

**Usage**

```
pin_date(pin, allow.temp = FALSE)
```

```
hetu_date(pin, allow.temp = FALSE)
```

**Arguments**

pin	Finnish personal identity code(s) as a character vector
allow.temp	Allow artificial or temporary PINs (personal numbers 900-999). If FALSE (default), only PINs intended for official use (personal numbers 002-899) are allowed.

**Value**

Date of birth as a vector in date format.

**Examples**

```
pin_date(c("010101-0101", "111111-111C"))
```

```
hetu_date(c("010101-0101", "111111-111C"))
```

---

pin_sex	<i>Extract Sex from Personal Identity Code</i>
---------	--

---

**Description**

Extract sex (as binary) from Finnish personal identification code.

**Usage**

```
pin_sex(pin, allow.temp = TRUE)
```

```
hetu_sex(pin, allow.temp = TRUE)
```

**Arguments**

`pin` Finnish personal identity code(s) as a character vector

`allow.temp` Allow artificial or temporary PINs (personal numbers 900-999). If FALSE (default), only PINs intended for official use (personal numbers 002-899) are allowed.

**Value**

Factor with label 'Male' and 'Female'.

**Author(s)**

Pyry Kantanen, Leo Lahti

**See Also**

[hetu](#) For general information extraction

**Examples**

```
pin_sex("010101-010A")
hetu_sex("010101-010A")
```

---

`plot.diagnostic` *Plotting method for diagnostic class objects*

---

**Description**

Creates a concise plot that visualizes TRUE and FALSE cases in a diagnostics data frame

**Usage**

```
## S3 method for class 'diagnostic'
plot(x, negate.logicals = FALSE, labels = TRUE, ...)
```

**Arguments**

`x` a "summary.diagnostic" object

`negate.logicals` negate TRUE and FALSE logicals, default is FALSE. Sometimes it may be beneficial to emphasize FALSE cases instead of TRUE

`labels` include column labels on y-axis, default is TRUE

`...` Arguments to be passed to methods, such as graphical parameters. For example:

- **type** what type of plot should be drawn. Default is "h" for histogram / high density vertical lines.
- **lwd** line width as double Default is 1.0

See [plot](#) and [par](#) for more options

## Details

There seems to be no canonical answer on what to call this type of plot. Some of the names that can be found online when describing a plot for binary response value on an axis are: a one-dimensional scatterplot, a sparkline, a rug plot, or a strip plot / strip chart.

---

rbid	<i>Generate Random Finnish Business ID's (Y-tunnus)</i>
------	---

---

## Description

A function that generates random Finnish Business ID's, bid-numbers (Y-tunnus).

## Usage

```
rbid(n)
```

## Arguments

n	number of generated BIDs
---	--------------------------

## Value

a vector of generated BID-numbers.

## Examples

```
x <- rbid(3)
bid_ctrl(x)
```

---

rpin	<i>Generate Random Personal Identity Codes</i>
------	--

---

## Description

A function that generates random Finnish personal identity codes (hetu codes).

## Usage

```
rpin(  
  n,  
  start.date = as.Date("1895-01-01"),  
  end.date = Sys.Date(),  
  p.male = 0.4,  
  p.temp = 0,  
  num.cores = 1  
)  
  
rhetu(  
  n,  
  start.date = as.Date("1895-01-01"),  
  end.date = Sys.Date(),  
  p.male = 0.4,  
  p.temp = 0,  
  num.cores = 1  
)
```

## Arguments

n	number of generated hetu-pins
start.date	Lower limit of generated hetu dates, character string in ISO 8601 standard, for example "2001-02-03". Default is "1895-01-01".
end.date	Upper limit of generated hetu. Default is current date.
p.male	Probability of males, between 0.0 and 1.0. Default is 0.4.
p.temp	Probability of temporary identification numbers, between 0.0 and 1.0. Default is 0.0.
num.cores	The number of cores for parallel processing. The number of available cores can be determined with <code>detectCores()</code> . Default is 1.

## Details

This function will return an error "too few positive probabilities" in `sample.int` function if you try to generate too many codes in a short enough timeframe. The theoretical upper limit of valid PINs is in the millions, but the number of valid PINs per day used to be 898 PINs at maximum, meaning 327770 for each year. Attempting to generate e.g. a 1000 pins for a timespan of one day would result in an error.

In practice this theoretical upper limit number was much lower since the old practice was that the same personal number component cannot be "recycled" if it has been used in the past. To illustrate, if an identity code "010101-0101" has already been assigned to someone born in 1901-01-01, a similar code "010101A0101" for someone born in 2001-01-01 could not be used.

In hetu package version 1.1.0 we have taken into account a new government decree that increased the amount of valid century markers and therefore increased the amount of valid personal codes per day. Additionally, the decree has made it possible to recycle individual codes, as the century marker is now thought to be a distinguishing character of the personal identity code.

However, the current implementation still keeps the old 898 codes per day limit intact, and assigns new century markers with a low probability: old markers "-" and "A" are given a 95 markers are given a 1

In the future this may be altered into a waterfall pattern so that the initial 898 codes for each date get "-" as the century marker, the next 898 get "Y", and so on. This would mean that each day would have 5388 valid codes and the distribution of century markers would be more realistic in the sense that additional century markers are taken into use only after the previous range has been exhausted. However, this would require generating rather large datasets even for basic testing purposes.

### Value

a vector of generated hetu-pins.

### Author(s)

Pyry Kantanen, Jussi Paananen

### Examples

```
x <- rpin(3)
hetu(x)
hetu(x, extract = "sex")
hetu(x, extract = "ctrl.char")

x <- rhetu(3)
x
```

---

satu\_control\_char      *Finnish Unique Identification Number Control Character Calculator*

---

### Description

Calculate a valid control character for an incomplete Finnish Unique Identification Number (FINUID, or sähköinen asiointitunnus SATU).

### Usage

```
satu_control_char(pin, print.full = FALSE)
```

### Arguments

pin	An incomplete FINUID that has 8 first numbers.
print.full	Should the function print only the whole FINUID-number (TRUE) or only the control character (FALSE). Default is FALSE.

**Details**

This method of calculating the control character was devised by mathematician Erkki Pale (1962) to detect input errors but also to detect errors produced by early punch card machines. The long number produced by writing the birth date and the personal number together are divided by 31 and the remainder is used to look up the control character from a separate table containing alphanumeric characters except letters G, I, O, Q and Z.

The method of calculating the control character does not need century character and therefore the function has an option to omit it.

**Value**

Control character, either a number 0-9 or a letter (length 1 character). If parameter `print.full` is set to `TRUE`, the function returns a complete FINUID / SATU number (length 9 characters).

**Author(s)**

Pyry Kantanen

**See Also**

For more detailed information about FINUID, see Finnish Digital and population data services agency website: <https://dvv.fi/en/citizen-certificate-and-electronic-identity>

**Examples**

```
# The first assigned FINUID number, 10000001N.
satu_control_char("10000001")
```

---

satu\_ctrl

*Check Validity of Finnish Unique Identification Number (SATU)*

---

**Description**

A function that checks whether a `satu` (Finnish Unique Identification Number) is valid. Returns `TRUE` or `FALSE`.

**Usage**

```
satu_ctrl(satu)
```

**Arguments**

`satu` a vector of 1 or more Unique Identification Numbers

**Examples**

```
satu_ctrl("10000001N") # TRUE
satu_ctrl(c("10000001N", "20000001B")) # TRUE FALSE
```

# Index

bid\_ctrl, 2

hetu, 3, 6, 7, 9, 11

hetu\_age (pin\_age), 8

hetu\_control\_char, 5

hetu\_ctrl (pin\_ctrl), 9

hetu\_date (pin\_date), 10

hetu\_diagnostic, 6

hetu\_sex (pin\_sex), 10

is.diagnostic, 7

par, 11

pin\_age, 8

pin\_ctrl, 5, 9

pin\_date, 10

pin\_diagnostic (hetu\_diagnostic), 6

pin\_sex, 10

plot, 11

plot.diagnostic, 11

rbid, 12

rhetu, 5

rhetu (rpin), 12

rpin, 12

sample.int, 13

satu\_control\_char, 14

satu\_ctrl, 15