METHODS AND APPARATUSES FOR LOCATING CODE SNIPPET, COMPUTER DEVICES AND STORAGE MEDIUMS

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to Chinese Patent Application No. 2019102961059, filed with the Chinese Patent Office on April 12, 2019 and entitled "METHODS AND APPARATUSES FOR LOCATING CODE SNIPPET, COMPUTER DEVICES AND STORAGE MEDIUMS", which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The present application relates to methods and apparatuses for locating a code snippet, computer devices, and storage mediums.

BACKGROUND

[0003] With the development of internationalization, the developed software systems will face the needs of being used in different regions and countries in the world, it is therefore necessary to translate the source code of the software system from the original language to a certain language. In the traditional way, it is usually the business analyst who collects translation requirements and writes the same into a document, and then delivers it to the programmer to modify the source code based on the document. However, the inventors realized that the conventional translation process of code internationalization requires the cooperation between the business analyst and the developer, the time cost and the manpower cost are high, and the software developer needs the software developer to manually look up the code snippet that needs translation from the large-scale source code, the workload to locate the code snippet to be translated from the source code is heavy and the checking is very difficult, code snippet omissions are likely to happen, it is therefore difficult to effectively locate the code snippet.

SUMMARY

[0004] According to various embodiments disclosed in this application, there is provided methods and apparatuses for locating a code snippet, computer devices, and storage mediums.

[0005] A method for locating a code snippet applied to a code server, the method includes:

[0006] acquiring a source code file to be translated;

[0007] reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

[0008] traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matches a preset language type corresponding to a terminal page display;

[0009] looking up a parser matching the preset language type; and

[0010] parsing the sub-file of code to be translated by the parser to obtain an analysis result file; the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

[0011] An apparatus for locating a code snippet includes:

[0012] an acquiring module configured to acquire a source code file to be translated;

[0013] a reading module configured to read a source code directory in the source code file to be translated; the source code directory includes a plurality of initial file names;

[0014] a filtering module configured to, based on the source code directory, acquire a file object corresponding to each initial file name, and write the file object into a file array; traverse the file array and, when the file object is an initialization code sub-file, filter out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matches a preset language type corresponding to a terminal page

display;

[0015] a looking up module configured to look up a parser matching the preset language type; and

[0016] a parsing module configured to parse the sub-file of code to be translated by the parser to obtain an analysis result file; the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

[0017] A computer device includes a memory and one or more processors, the memory storing computer-readable instructions that, when executed by the one or more processors, causing the one or more processors to perform the following steps:

[0018] acquiring a source code file to be translated;

[0019] reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

[0020] traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matches a preset language type corresponding to a terminal page display;

[0021] looking up a parser matching the preset language type; and

[0022] parsing the sub-file of code to be translated by the parser to obtain an analysis result file; the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

[0023] One or more non-volatile computer-readable storage medium storing computer-readable instructions that, when executed by one or more processors, causing the one or more processors to perform the following steps:

[0024] acquiring a source code file to be translated;

[0025] reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

[0026] traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matches a preset language type corresponding to a terminal page display;

[0027] looking up a parser matching the preset language type; and

[0028] parsing the sub-file of code to be translated by the parser to obtain an analysis result file; the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

[0029] The details of at least an embodiment of the present disclosure will be presented with reference to the following drawings and description. Other characteristic and advantages of the present disclosure will be more apparent from the specification, drawings and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0030] To illustrate the technical solutions according to the embodiments of the present disclosure more clearly, the accompanying drawings for describing the embodiments are introduced briefly in the following. Apparently, the accompanying drawings in the following description are only some embodiments of the present disclosure, and persons of ordinary skill in the art can derive other drawings from the accompanying drawings without creative efforts.

[0031] FIG. 1 is a schematic diagram of an application scenario of a method for locating a code snippet according to one or more embodiments.

[0032] FIG. 2 is a schematic flowchart of a method for locating a code snippet according to one or more embodiments.

[0033] FIG. 3 is a schematic flowchart of a method for locating a code snippet according to another embodiment.

[0034] FIG. 4 is a block diagram of an apparatus for locating a code snippet according to one or more embodiments.

[0035] FIG. 5 is a schematic diagram of The computer device according to one or

more embodiments.

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0036] The present disclosure will be described in details in combination with the accompanying drawings and embodiments such that the technical solution and advantages of the present disclosure will be more apparent. It should be understood that the particular embodiments are described for the purpose of illustrating as opposed to restricting the present disclosure.

[0037] The method for locating a code snippet provided in this application may be applied to the application environment shown in FIG. 1. The code server 102 communicates with the translation server 104 via the network. The code server 102 and translation server 104 can be implemented by a separate server or a cluster of servers composed of a plurality of servers. After acquiring the source code file to be translated, the code server 102 can read the source code directory in the source code file to be translated. The code server 102 traverses the initial code sub-files corresponding to a plurality of initial file names included in the source code directory to filter and obtain the sub-file of code to be translated matching the preset language type. By looking up the parser matching the preset language type and parsing the sub-file of code to be translated, the code server 102 can obtain the analysis result file including the code snippet to be translated and the corresponding snippet locating information. The code server 102 may also send the analysis result file to the translation server 104 for translation processing in the form of a data stream.

[0038] In an embodiment, as shown in FIG. 2, a method for locating a code snippet is provided and illustrated being applied to the code server 102 in FIG. 1, which includes the following steps:

[0039] At step 202, a source code file to be translated is acquired.

[0040] The source code file to be translated refers to the source code file that needs to be translated. Source code files are uncompiled text files written in accordance with certain programming language specifications. The source code to be translated can be pre-stored in the database of the code server, the code server can directly

acquire the source code files to be translated from the local database. The code server can also receive to-be-translated code files uploaded by terminals or source code files to be translated sent by other servers.

[0041] At step 204, a source code directory in the source code file to be translated is read; the directory of the source code to be translated includes a plurality of initial file names.

[0042] After reading the configuration file of the source code file to be translated, the configuration parameters in the configuration file can be pre-checked. After the pre-checking is passed, the source code directory in the configuration file can be read. The configuration file stores the source code directory generated by the initial file names of a plurality of initial code sub-files. The source code directory can also be a hierarchical structure, the source code directory can also include subdirectories, so that the source code directory can all be expanded, all the file names listed in the expanded source code directory are regarded as initial file names.

[0043] At step 206, an initial code sub-file corresponding to each of the initial file names is traversed, and a sub-file of code to be translated is filtered out; the sub-file of code to be translated matches a preset language type.

[0044] The initial code sub-file refers to the source code file in the source code file to be translated that is of the smallest granularity and cannot be further expanded. The sub-file of code to be translated refers to the initial code sub-file matching the preset language type and needs to be translated into the target language type. The sub-file of code to be translated includes the code snippet to be translated. In order to save system resources, only the to-be-translated sub-files including the code snippet to be translated are filtered out to avoid futile processing on the initial code sub-files that need not to be translated.

[0045] In an embodiment, a file object corresponding to each initial file name is acquired based on the source code directory, and the file object is placed in the file array. By traversing the file array and, when the file object is a subdirectory, the file object corresponding to the subdirectory is expanded recursively; when the file object is an initial code sub-file, the preset language type matching the initial code sub-file is

directly detected.

[0046] In an embodiment, when it is detected that the initial code sub-file does not match any of the preset language types, the initial code sub-file may be marked as processed and the initial file name of the initial code sub-file is written to the analysis result file.

[0047] At step 208, the parser matching the preset language type is looked up.

[0048] A plurality of parsers corresponding to a variety of preset language types can be pre-installed in the code server, and the sub-file of code to be translated is processed by matched parsers. The code server may also call parsers deployed on other computer devices, for example, a parser corresponding to a preset language type may be pre-installed on each parsing server, and the code server may send the to-be-translated sub-file to the corresponding parsing server for processing according to the matched preset language type. The preset language types include but are not limited to one or more of Chinese, Thai, Burmese, Cambodian and Laos. The parser can be a standalone application or a plug-in running on a specific system platform. For example, the parser can be Babel parser (a JavaScript source parser).

[0049] In an embodiment, different parsers may be preset for sub-file of codes to be translated of different file format types. After detecting the file name suffix of the sub-file of code to be translated, a plurality of parser sets corresponding to the file format type of the sub-file of code to be translated can be looked up based on the file format type corresponding to the file name suffix of the sub-file of code to be translated, and a parser matching the preset language type of the sub-file of code to be translated can be looked up from the parser set. For example, different parsers can be pre-installed for sub-files of code to be translated with suffixes of ".js" (JavaScript) and ".jsx" (JavaScript XML, a XML-like grammar that organize internal build tag at React)).

[0050] In an embodiment, when a parser matching a preset language type is not found, the initial code sub-file may be marked as unprocessed and the initial file name of the initial code sub-file is written to the analysis result file.

[0051] At step 210, the sub-file of code to be translated is parsed by the parser to

obtain an analysis result file; the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

[0052] By parsing the sub-file of code to be translated using the parser, a syntax analysis can be performed on the to-be-processed code in the sub-file of code to be translated, and the code snippet to be translated in the to-be-processed code can be obtained. According to the code snippet to be translated obtained through analysis, the snippet locating information of the code snippet to be translated in the to-be-processed code is divided. For example, the code snippet to be translated of a specific syntax corresponds to a specific start string and end string, which can detect the start locating information of the start string in the to-be-processed code and the end locating information of the end string in the to-be-processed code, and the start locating information and the end locating information are regarded as the snippet locating information. A plurality of code snippets may also be divided and located in the to-be-processed code in advance, such as a code in the to-be-processed code and of a preset character length is regarded as a code snippet, the code snippet may be regarded as a code snippet to be translated when a specific string is detected in the code snippet.

[0053] In the described method for locating a code snippet, after acquiring the source code file to be translated, the code server can read the source code directory in the source code file to be translated. By traversing the initial code sub-files corresponding to the plurality of initial file names included in the source code directory, the sub-files of code to be translated matching the preset language type is filtered and obtained. After looking up the parser matching the preset language type, the sub-file of code to be translated can be parsed using the parser to obtain the analysis result file including the code snippet to be translated and the respective snippet locating information. By identifying and filtering out the sub-file of code to be translated matching the preset language type, the sub-file of code to be translated can be parsed more targeted according to the matched parser, so that and the code snippet to be translated in the sub-file of code to be translated can be automatically

and efficiently checked and located.

[0054] In an embodiment, the traversing the initial code sub-file corresponding to each initial file name and filtering out the sub-file of code to be translated includes: acquiring regular expressions respectively corresponding to different preset language types; traversing the file array, acquiring the initial code sub-file corresponding to each initial file name, and matching each initial code sub-file by the regular expression; determining the preset language type matching each initial code sub-file according to the matched regular expression; and filtering out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

[0055] The regular expression is a logical equation for a string operation, that is, to compose a "rule string" using specific characters and the combinations thereof, the "rule string" can be configured to express a filter logic for a string. Each preset language type has one or more respective regular expressions. Through regular expressions, it is possible to learn whether there are any code snippets to be translated in the code to be processed, so as to avoid parsing the initial code sub-files that do not include the code snippet to be translated, thereby reducing the parsing workload and improving the parsing efficiency.

[0056] In an embodiment, parsing the sub-file of code to be translated using the parser to obtain the analysis result file includes detecting the code snippet to be translated in the sub-file of code to be translated; dividing a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; the snippet locating information includes start location information and end location information; extracting the code snippet to be translated based on the start location information and the end location information; and generating the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

[0057] The analysis result file is the result file generated by subjecting the code in the whole source code file to be translated to a language type analysis and a syntax analysis by the parser. Based on the analysis, the code snippet to be translated, the start location information, and the end location information are extracted to generate

the analysis result file. Based on the analysis result file, the code snippet to be translated can be accurately located in the source code file to be translated, thereby allowing further code translation processing and code replacement processing to be performed according to the analysis result file.

[0058] In an embodiment, the detecting the code snippet to be translated in the sub-file of code to be translated includes acquiring a template string; matching the template string with the sub-file of code to be translated to determine an occupation location information of the template string in the sub-file of code to be translated; and determining the code snippet to be translated according to the occupation location information of the template string.

[0059] The code snippets to be translated can be one or more types of code snippets, such as a JS String type, a ES6 template type (ECMAScript 6, template string), a JSX text type (JavaScript XML, a XML-like grammar that organize internal build tag at React). For example, the template string may be a string corresponding to a string object attribute, such as "constructor", "length", etc. the template string can also be a string corresponding to string object methods, such as "anchor ()", "fontcolor ()", and "link ()". Since these types of code snippets involve lots of terminal page displays, these types of code snippets need to be regarded as code snippets to be translated.

[0060] In an embodiment, in addition to filtering out the code snippets to be translated, the code snippets that need not to be translated can be marked. For example, through template string matching, detecting if/switch determination condition type, and split type and other code snippets that involve strong business logic, or detecting code annotation, console output and other code snippets that are parts of no effect, by marking the code snippets that need not to be translated, the developers can check and distinguish subsequently with ease. It is also possible to distinguish between different types of code snippets by marking different colors so that code snippets that require different processing can be distinguished intuitively.

[0061] In an embodiment, after the extracting the code snippet to be translated based on the start location information and the end location information, the method

further includes: based on the start location information and the end location information, acquiring a code snippet of a preset character length before and after the code snippet to be translated as a context code snippet; parsing the code snippet to be translated with the context code snippet to obtain a syntax information matching the code snippet to be translated; and adding the context code snippet and the syntax information to the analysis result file.

[0062] For example, it is possible to acquire the code snippets of the first and last 50 character length of the code snippet to be translated as the context code snippet of the code snippet to be translated. With the context code snippet, the syntax analysis can be performed on the code snippet to be translated more accurately, and the developer can understand the location information of the code snippet to be translated more intuitively. Therefore, the code snippet to be translated can be re-divided based on the context code snippet to avoid the incompleteness of the code snippet. The start location information and end location information of the code snippet to be translated can be redetermined according to the code snippet to be translated after redivision.

[0063] In an embodiment, after the parsing the sub-file of code to be translated by the parser to obtain the analysis result file, further including: generating a plurality of data streams according to the analysis result file; sending the plurality of data streams to the translation server in a parallel transmission manner for translation processing; receiving a translated code snippet returned by the translation server; and replacing the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.

[0064] The analysis result file can include a plurality of code snippets to be translated and snippet locating information corresponding to each code snippet to be translated. Each code snippet to be translated and the snippet locating information corresponding to the respective code snippet to be translated can be regarded as an analysis result. Each code snippet to be translated and its corresponding snippet locating information can generate a data stream, and a plurality of code snippets to be translated can correspond to a data stream respectively. A data stream can also be

generated by synthesizing a preset number of code snippets to be translated and their corresponding snippet locating information, and a preset number of code snippets to be translated can correspond to a data stream. The snippet locating information includes but is not limited to a path of source code file to be translated and the location information of the code snippet to-be-translated in the respective sub-file of code to be translated. The analysis results may also include other relevant information corresponding to each code snippet to be translated, including but not limited to one or more of the context code snippet of the code snippet to be translated, the corresponding code syntax type obtained after parsing the code snippet to be translated, and the code syntax type of the parent node of the code to-be-translated.

[0065] After forming the data stream, the code server can perform a condition matching on each analysis result to look up the translation server matching each snippet to be translated. The translation server can have a code processor configured, and the context code snippet of the snippet to be translated satisfies the selection condition of the translation server. The translation server processes each analysis result through the code processor to identify the target language type the source code file to be translated needed to be translated into; the translation server calls the translator through the interface to obtain the translation result of the target language type and corresponding to each code snippet to be translated; a translated code snippet is generated according to the translation results; the translation server can return the translated code snippet to the code server. The code server can replace the code snippet to be translated in the source code file to be translated with the translated code snippet information based on snippet locating information such as the path of source code file to be translated and a location information of the code snippet to be translated in the source code file to be translated, so as to implement the translation of the source code file to be translated; and the translated source code file corresponding to the target language type is outputted after translation.

[0066] In an embodiment, as shown in FIG. 3, a method for locating a code snippet is provided and illustrated being applied to the code server 102 in FIG. 1, which includes the following steps:

[0067] At step 302, a source code file to be translated is acquired.

[0068] At step 304, a source code directory in the source code file to be translated is read; the directory of the source code to be translated includes a plurality of initial file names.

[0069] At step 306, regular expressions respectively corresponding to a plurality of preset language types are acquired.

[0070] At step 308: the initial code sub-file corresponding to each initial file name are traversed, and each initial code sub-file is matched with a regular expression.

[0071] At step 310, the preset language type matching each initial code sub-file is determined according to the matched regular expression.

[0072] At step 312, the initial code sub-file including any of the preset language types is filtered out as the sub-file of code to be translated.

[0073] At step 314, a parser matching the preset language type is looked up.

[0074] At step 316, the code snippet to be translated in the sub-file of code to be translated is detected by the parser.

[0075] At step 318, a snippet locating information of the code snippet to be translated in the sub-file of code to be translated is divided; the snippet locating information includes start location information and end location information.

[0076] At step 320, the code snippet to be translated is extracted based on the start location information and the end location information.

[0077] At step 322, the analysis result file is generated according to the code snippet to be translated, the start location information, and the end location information.

[0078] In the described method for locating a code snippet, after acquiring the source code file to be translated, the code server can read the source code directory in the source code file to be translated. By traversing the initial code sub-files corresponding to a plurality of initial file names included in the source code directory through a preset regular expression, a sub-file of code to be translated matching the preset language type can be filtered out and obtained. After looking up the parser matching the preset language type, the sub-file of code to be translated can be parsed

by the parser to obtain the analysis result file which includes the respective snippet locating information of the code snippet to be translated. By filtering out the to-be-translated sub-file matching the preset language type, the sub-file of code to be translated can be parsed more targeted according to the matched parser, so that the to-be-translated sub-file can be checked and located automatically, accurately and efficiently.

[0079] It should be understood that although the steps in the flow charts of FIGS. 2 and 3 are shown sequentially as indicated by arrows, these steps are not necessarily performed sequentially as indicated by arrows. Unless expressly stated in this article, these steps are not implemented in strict order, and they can be executed in other order. In addition, at least some of the steps in FIGS. 2 and 3 may include a plurality of sub-steps or stages that are not necessarily performed at a same time, but may be performed at different times. The sequence of execution of these sub-steps or stages is not necessarily performed in sequence, but may be performed alternately or alternately with at least some of the sub-steps or stages of other steps or other steps.

[0080] In an embodiment, as shown in FIG. 4, an apparatus for locating a code snippet 400 is provided, which includes an acquiring module 402 configured to acquire a source code file to be translated; a reading module 404 configured to read the source code directory in the source code file to be translated; the source code directory includes a plurality of initial file names; a filtering module 406 configured to traverse the initial code sub-file corresponding to each initial file name and filter out a sub-file of code to be translated; the sub-file of code to be translated matches a preset language type; a looking up module 408 configured to look up a parser matching the preset language type; a parsing module 410 configured to parse the sub-file of code to be translated by the parser to obtain the analysis result file; the analysis result file includes the code snippets to be translated in the sub-file of code to be translated and the locating information of the pieces corresponding to the code snippets to be translated.

[0081] In an embodiment, the filtering module 406 is further configured to acquire

regular expressions respectively corresponding to different preset language types; traverse the file array, acquire the initial code sub-file corresponding to each initial file name, and match each initial code sub-file by the regular expression; determine the preset language type matching each initial code sub-file according to the matched regular expression; and filter out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

[0082] In an embodiment, the parsing module 410 is further configured to detect the code snippet to be translated in the sub-file of code to be translated; divide a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; the snippet locating information includes start location information and end location information; extract the code snippet to be translated based on the start location information and the end location information; and generate the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

[0083] In an embodiment, the parsing module 410 is further configured to acquire a template string according to the preset language type corresponding to the terminal page display; match the template string with the sub-file of code to be translated to determine an occupation location information of the template string in the sub-file of code to be translated; and determine the code snippet to be translated according to the occupation location information of the template string.

[0084] In an embodiment, the parsing module 410 is further configured to acquire a code snippet of a preset character length before and after the code snippet to be translated as a context code snippet; parse the code snippet to be translated with the context code snippet to obtain a syntax information matching the code snippet to be translated; and add the context code snippet and the syntax information to the analysis result file.

[0085] In an embodiment, the apparatus further includes a translation module configured to generate a plurality of data streams according to the analysis result file; send the plurality of data streams to the translation server in a parallel transmission manner for translation processing; receive a translated code snippet returned by the

translation server; and replace the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.

[0086] For specific definition of the apparatus for locating a code snippet, reference may be made to the definition of the above method for locating a code snippet, and details are omitted for brevity. Each of the modules in the apparatus for locating a code snippet described may be entirely or partially implemented by software, hardware, or a combination thereof. Each of the modules may be embedded in or independent of the processor in the computer device in the form of hardware, or may be stored in the memory in the computer device in the form of software, so that the processor calls and executes the operations corresponding to each of the modules.

[0087] In an embodiment, a computer device is provided, which may be a code server and an internal structure thereof may be as shown in FIG. 5. The computer device includes a processor, a memory, a network interface, and a database connected through a system bus. The processor of the computer device is configured to provide calculation and control capabilities. The memory of the computer device includes a non-transitory storage medium and a random access memory (RAM). The non-transitory storage medium stores an operating system, computer-readable instructions and the database. The RAM provides an environment for the operation of the operating system and the computer-readable instructions in the non-transitory storage medium. The database of the computer device is configured to store data such as the to-be-translated analysis result set. The network interface of the computer device is configured to communicate with an external terminal through a network connection. The computer-readable instructions are executed by the processor to implement the method for locating a code snippet.

[0088] Those skilled in the art will appreciate that the structure shown in FIG. 5 is merely a block diagram of a portion of the structure associated with the solution of the present disclosure, and does not constitute a limitation on the computer device to which the solution of the present disclosure is applied, a particular computer device may include more or less components, or combine certain components, or with a

different arrangement of components.

[0089] In an embodiment, a computer device is provided, which includes a memory and one or more processors, the memory store computer-readable instructions that, when executed by the one or more processors, cause the one or more processors to perform the steps of the method for locating a code snippet according to any of the foregoing embodiments.

[0090] In an embodiment, one or more non-volatile computer-readable storage media is provided, which stores computer-readable instructions that, when executed by one or more processors, causing the one or more processors to perform the steps of the method for locating a code snippet according to any of the foregoing embodiments.

[0091] Persons of ordinary skill in the art understand that all or part of the processes in the methods of the foregoing embodiments may be implemented by a computer program instructing relevant hardware. The computer program may be stored in a non-transitory computer-readable storage medium. When the computer program is executed, flows of embodiments of the methods as described above may be included. Any references to memory, storage, databases, or other media used in the various embodiments provided herein may include non-transitory and/or transitory memory. The non-transitory memory may include a read only memory (ROM), a programmable ROM (PROM), an electrically programmable ROM (EPROM), an electrically erasable programmable ROM (EEPROM), or a flash memory. The transitory memory may include a random access memory (RAM) or an external cache memory. By way of illustration and not limitation, RAM is available in a variety of forms such as static RAM (SRAM), dynamic RAM (DRAM), synchronous DRAM (SDRAM), double data rate SDRAM (DDRSDRAM), enhanced SDRAM (ESDRAM), Synchlink DRAM (SLDRAM), Memory Bus (Rambus) Direct RAM (RDRAM), Direct Memory Bus Dynamic RAM (DRDRAM), Memory Bus Dynamic RAM (RDRAM) etc.

[0092] The foregoing respective technical features involved in the respective embodiments can be combined arbitrarily, for brevity, not all possible combinations

of the respective technical features in the foregoing embodiments are described, however, to the extent they have no collision with each other, the combination of the respective technical features shall be considered to be within the scope of the description.

[0093] The foregoing implementations are merely specific embodiments of the present disclosure, and are not intended to limit the protection scope of the present disclosure. It should be noted that any variation or replacement readily figured out by persons skilled in the art within the technical scope disclosed in the present disclosure shall all fall into the protection scope of the present disclosure. Therefore, the protection scope of the present disclosure shall be subject to the protection scope of the claims.

What is claimed is:

1. A method for locating a code snippet applied to a code server, the method comprising:

acquiring a source code file to be translated;

reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matching a preset language type corresponding to a terminal page display;

looking up a parser matching the preset language type; and

parsing the sub-file of code to be translated by the parser to obtain an analysis result file; wherein the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

2. The method according to claim 1, wherein the traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file comprises:

acquiring regular expressions respectively corresponding to different preset language types;

traversing the file array, acquiring the initial code sub-file corresponding to each initial file name, and matching each initial code sub-file by the regular expression;

determining the preset language type matching each initial code sub-file according to the matched regular expression; and

filtering out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

3. The method according to claim 1, wherein the parsing the sub-file of code to be translated by the parser to obtain the analysis result file comprises:

detecting the code snippet to be translated in the sub-file of code to be translated;

dividing a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; wherein the snippet locating information incudes start location information and end location information;

extracting the code snippet to be translated based on the start location information and the end location information; and

generating the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

4. The method according to claim 3, wherein the detecting the code snippet to be translated in the sub-file of code to be translated comprises:

acquiring a template string according to the preset language type corresponding to the terminal page display;

matching the template string with the sub-file of code to be translated to determine an occupation location information of the template string in the sub-file of code to be translated; and

determining the code snippet to be translated according to the occupation location information of the template string.

5. The method according to claim 3, wherein after the extracting the code snippet to be translated based on the start location information and the end location information, the method further comprising:

based on the start location information and the end location information, acquiring a code snippet of a preset character length before and after the code snippet to be translated as a context code snippet;

parsing the code snippet to be translated with the context code snippet to obtain a syntax information matching the code snippet to be translated; and

adding the context code snippet and the syntax information to the analysis result file.

6. The method according to any one of claims 1 to 5, wherein after the parsing the sub-file of code to be translated by the parser to obtain the analysis result file, the method further comprising:

generating a plurality of data streams according to the analysis result file; performing condition matching on each analysis result in the analysis result file and looking up a translation server matching each snippet to be translated; a code processor being deployed on the translation server;

sending the plurality of data streams to the translation server in a parallel transmission manner, the translation server performs translation processing on an analysis result in the data stream using the code processor;

receiving a translated code snippet returned by the translation server; and

replacing the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.

7. An apparatus for locating a code snippet, comprising:

an acquiring module configured to acquire a source code file to be translated;

a reading module configured to read a source code directory in the source code file to be translated; the source code directory comprising a plurality of initial file names;

a filtering module configured to, based on the source code directory, acquire a file object corresponding to each initial file name, and write the file object into a file array; traverse the file array and, when the file object is an initialization code sub-file, filter out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matching a preset language type corresponding to a terminal page display;

a looking up module configured to look up a parser matching the preset

language type; and

a parsing module configured to parse the sub-file of code to be translated by the parser to obtain an analysis result file; wherein the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

8. The apparatus according to claim 7, wherein the filtering module is further configured to acquire regular expressions respectively corresponding to different preset language types; traverse the file array, acquire the initial code sub-file corresponding to each initial file name, and match each initial code sub-file by the regular expression; determine the preset language type matching each initial code sub-file according to the matched regular expression; and filter out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

9. The apparatus of claim 7, wherein the parsing module is further configured to detect the code snippet to be translated in the sub-file of code to be translated; divide a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; wherein the snippet locating information includes start location information and end location information; extract the code snippet to be translated based on the start location information and the end location information; and generate the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

10. The apparatus according to any one of claims 7 to 9, wherein the apparatus further comprises a translation module configured to generate a plurality of data streams according to the analysis result file; perform condition matching on each analysis result in the analysis result file and looking up a translation server matching each snippet to be translated; a code processor being deployed on the translation server; send the plurality of data streams to the translation server in a parallel transmission

manner, the translation server performs translation processing on an analysis result in the data stream using the code processor; receive a translated code snippet returned by the translation server; and replace the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.

11. A computer device comprising a memory and one or more processors, the memory storing computer-readable instructions that, when executed by the one or more processors, causing the one or more processors to perform the following steps:

acquiring a source code file to be translated;

reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matching a preset language type corresponding to a terminal page display;

looking up a parser matching the preset language type; and

parsing the sub-file of code to be translated by the parser to obtain an analysis result file; wherein the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

12. The computer device of claim 11, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

acquiring regular expressions respectively corresponding to different preset language types;

traversing the file array, acquiring the initial code sub-file corresponding to each initial file name, and matching each initial code sub-file by the regular

expression;

determining the preset language type matching each initial code sub-file according to the matched regular expression; and

filtering out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

13. The computer device of claim 11, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

detecting the code snippet to be translated in the sub-file of code to be translated;

dividing a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; wherein the snippet locating information includes start location information and end location information;

extracting the code snippet to be translated based on the start location information and the end location information; and

generating the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

14. The computer device of claim 13, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

acquiring a template string according to the preset language type corresponding to the terminal page display;

matching the template string with the sub-file of code to be translated to determine an occupation location information of the template string in the sub-file of code to be translated; and

determining the code snippet to be translated according to the occupation location information of the template string.

15. The computer device of claim 13, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

based on the start location information and the end location information, acquiring a code snippet of a preset character length before and after the code snippet to be translated as a context code snippet;

parsing the code snippet to be translated with the context code snippet to obtain a syntax information matching the code snippet to be translated; and

adding the context code snippet and the syntax information to the analysis result file.

16. The computer device according to any one of claims 11 to 15, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

generating a plurality of data streams according to the analysis result file; performing condition matching on each analysis result in the analysis result file and looking up a translation server matching each snippet to be translated; a code processor being deployed on the translation server;

sending the plurality of data streams to the translation server in a parallel transmission manner, the translation server performs translation processing on an analysis result in the data stream using the code processor;

receiving a translated code snippet returned by the translation server; and

replacing the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.

17. One or more non-volatile computer-readable storage medium storing computer-readable instructions that, when executed by one or more processors, causing the one or more processors to perform the following steps:

acquiring a source code file to be translated;

reading a source code directory in the source code file to be translated; based on the source code directory, acquiring a file object corresponding to each initial file name, and writing the file object into a file array;

traversing the file array and, when the file object is an initialization code sub-file, filtering out a sub-file of code to be translated in the initial code sub-file; the sub-file of code to be translated matching a preset language type corresponding to a terminal page display;

looking up a parser matching the preset language type; and

parsing the sub-file of code to be translated by the parser to obtain an analysis result file; wherein the analysis result file includes a code snippet to be translated in the sub-file of code to be translated and snippet locating information corresponding to the code snippet to be translated.

18. The computer readable storage medium of claim 17, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

acquiring regular expressions respectively corresponding to different preset language types;

traversing the file array, acquiring the initial code sub-file corresponding to each initial file name, and matching each initial code sub-file by the regular expression;

determining the preset language type matching each initial code sub-file according to the matched regular expression; and

filtering out the initial code sub-file including any of the preset language types as the sub-file of code to be translated.

19. The computer readable storage medium of claim 17, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

detecting the code snippet to be translated in the sub-file of code to be

translated;

dividing a snippet locating information of the code snippet to be translated in the sub-file of code to be translated; wherein the snippet locating information includes start location information and end location information;

extracting the code snippet to be translated based on the start location information and the end location information; and

generating the analysis result file according to the code snippet to be translated, the start location information, and the end location information.

20. The computer readable storage medium according to any one of claims 17 to 19, wherein when the computer-readable instructions are executed by one or more processors, further causes the one or more processors to perform the following steps:

generating a plurality of data streams according to the analysis result file; performing condition matching on each analysis result in the analysis result file and looking up a translation server matching each snippet to be translated; a code processor being deployed on the translation server;

sending the plurality of data streams to the translation server in a parallel transmission manner, the translation server performs translation processing on an analysis result in the data stream using the code processor;

receiving a translated code snippet returned by the translation server; and

replacing the respective code snippet to be translated in the sub-file of code to be translated according to the translated code sub-file to obtain the translated code sub-file.