

APP PROGRAM RUNNING METHOD AND APPARATUS

TECHNICAL FIELD

[0001] The present disclosure relates to the field of Internet technologies, and in particular, to an APP running method and apparatus.

5

BACKGROUND

[0002] In smart technology era, more and more applications (APPs) are used by users, and the APPs need to be downloaded and installed before using. Currently, installation package of the APP is usually large, for example, about 100 M. When a user downloads an installation package from an APP store, large disk space of a terminal device is occupied. In addition, large download traffic and long download time are needed, increasing resource overheads.

SUMMARY

[0003] The present disclosure aims to resolve one of technical problems in a related technology at least to a certain extent.

15 [0004] Therefore, an objective of the present disclosure is to provide an application (APP) running method, and the method can reduce the size of an APP installation package, reducing resource overheads.

[0005] Another objective of the present disclosure is to provide an APP running apparatus.

20 [0006] To achieve the earlier described objectives, an APP running method proposed in an implementation of a first aspect of the present disclosure includes: generating a class description file based on an original file of an APP, and placing the class description file in an installation package, where the class description file is a text file used for class description, and the size of the class description file is smaller than the size of a corresponding class; downloading the installation package before a user runs the APP; and when the APP runs, obtaining the class description file from

25

the installation package, and generating a class based on the class description file.

[0007] In the APP running method proposed in the implementation of the first aspect of the present disclosure, the installation package includes the class description file, and the size of the class description file is smaller than that of the class.

5 Therefore, compared with an existing method of including a class directly in an installation package, including the class description file in the installation package can reduce the size of the APP installation package, reducing resource overheads.

[0008] To achieve the earlier described objectives, an APP running apparatus proposed in an implementation of a second aspect of the present disclosure includes: a
10 generation module, configured to: generate a class description file based on an original file of an APP, and place the class description file in an installation package, where the class description file is a text file used for class description, and the size of the class description file is smaller than the size of a corresponding class; a download
module, configured to download the installation package before a user runs the APP;
15 and a running module, configured to: when the APP runs, obtain the class description file from the installation package, and generate a class based on the class description file.

[0009] In the APP running apparatus proposed in the implementation of the second aspect of the present disclosure, the installation package includes the class
20 description file, and the size of the class description file is smaller than that of the class. Therefore, compared with an existing method of including a class directly in an installation package, including the class description file in the installation package can reduce the size of the APP installation package, reducing resource overheads.

[0010] Some additional aspects and advantages of the present disclosure are
25 provided in the following descriptions, are apparent in the following descriptions, or are understandable through practice of the present disclosure.

BRIEF DESCRIPTION OF DRAWINGS

[0011] The earlier described and/or additional aspects and advantages of the present disclosure are apparent and comprehensible in the following descriptions of
30 implementations with reference to the accompanying drawings.

[0012] FIG. 1 is a schematic flowchart illustrating an APP running method, according to an implementation of the present disclosure;

[0013] FIG. 2 is a schematic diagram illustrating an APP running method in the existing technology;

[0014] FIG. 3 is a schematic flowchart illustrating an APP running method, according to another implementation of the present disclosure;

5 [0015] FIG. 4 is a schematic flowchart of generating a class based on a class description file, according to an implementation of the present disclosure;

[0016] FIG. 5 is a schematic structural diagram of an APP running apparatus, according to an implementation of the present disclosure; and

[0017] FIG. 6 is a schematic structural diagram of an APP running apparatus,
10 according to another implementation of the present disclosure.

DESCRIPTION OF EMBODIMENTS

[0018] The implementations of the present disclosure are described in detail below, and examples of the implementations are shown in the accompanying drawings. Same or similar reference numerals represent the same or similar modules
15 or modules having the same or similar functions. The implementations described below with reference to the accompanying drawings are examples, intended for explaining the present disclosure, and should not be construed as a limitation of the present disclosure. On the contrary, the implementations of the present disclosure include all changes, modifications and equivalents falling within the spirit and scope
20 of the appended claims.

[0019] FIG. 1 is a schematic flowchart illustrating an application (APP) running method, according to an implementation of the present disclosure. The method includes the following steps.

[0020] S11. Generate a class description file based on an original file of an APP,
25 and place the class description file in an installation package, where the class description file is a text file used for class description, and the size of the class description file is smaller than the size of a corresponding class.

[0021] For example, the original file includes a header file (.h file) and a source file (.m file).

30 [0022] The class description file is a text file and used to describe related information of a class. Classes are basic information in computer programming. The size of the class description file is smaller than the size of a corresponding class.

[0023] For example, a class name (className) is cashier (MERCHANTAPPOperator), and the cashier has a class description (classDescription). The class description includes attribute description, which for example, can include a cashier ID (operatorId), a cashier name (operatorName), a shop name (shopName) of the cashier, etc.

[0024] Using the iOS as an example, a pre-compiled script can be used to convert the original file into the class description file (.plist file). Content needed in the class description file can be found in the original file, but the original file and the class description file have different formats. Therefore, the script can be pre-compiled to implement conversion between the original file and the class description file.

[0025] In addition, after the class description file is generated, it can be stored in a predetermined directory, such as a resource bundle.

[0026] After the class description file is generated, it can be placed in the installation package for a user to download and install.

[0027] In some implementations, referring to FIG. 2, the generating a class description file based on an original file of an APP, and placing the class description file in an installation package includes the following steps.

[0028] S111. Generate the class description file based on a header file and a source file.

[0029] For a generation procedure, refer to the earlier described related description. Details are not described herein again.

[0030] S112. Delete the source file.

[0031] S113. After the source file is deleted, place the header file and the class description file in the installation package for the user to download.

[0032] Because the class description file and the header file are both small, and the obtained installation package is also small, the user can directly download the installation package from the APP store.

[0033] Optionally, to further reduce the size of the installation package, the obtained installation package can be further compressed. For example, after the header file and the class description file are included in the installation package, the method can further include the following steps.

[0034] S114. Compress the installation package, so that the user downloads a compressed installation package.

[0035] For example, the compressed installation package is presented in the APP

store, and the user can download the compressed installation package.

[0036] S12. Download the installation package before a user runs the APP.

[0037] For example, the user can download the installation package of the APP from the APP store, and the installation package includes the class description file.

5 **[0038]** To better distinguish the present disclosure from the existing technology, FIG. 3 shows a schematic flowchart of running an APP in the existing technology. Referring to FIG. 3, in the existing technology, an installation package is generated in two steps: compile a source file to generate an intermediate file (.o file) (S31), and then package a link of the intermediate file to generate the installation package (S32).
10 For example, in the iOS system, the installation package can be referred to as an ipa package. After the compiling and link packaging, the ipa package includes a class corresponding to an APP. Afterwards, the ipa package can be placed in an APP store for a user to download. After download, the ipa package is stored in a disk of a terminal device (S33). When the APP runs, the ipa package is read from the disk to a
15 memory of the terminal device for running (S34). Therefore, normal running of the APP is implemented. The ipa package is generally large, and therefore large amount of resources are needed for download and storage.

[0039] Different from the existing technology, compiling and link packaging of the original file are not needed in this implementation, class description file is used
20 instead. The class description file is a text file. Compared with the existing ipa package, the size of the class description file is small, so as to reduce resources for download and storage.

[0040] S13. When the APP runs, obtain the class description file from the installation package, and generate a class based on the class description file.

25 **[0041]** When the downloaded installation package is the compressed installation package, the installation package can first be decompressed when the installation packaged is used, so as to obtain information such as the class description file. When the downloaded installation package is not compressed, information such as the class description file can be directly obtained from the installation package.

30 **[0042]** To ensure normal running of the APP, the class included in the APP needs to be generated. In the existing technology, the class can be generated by compiling and packaging the source file, so that the ipa package stored in the disk includes the class. When the APP runs, the class can be directly obtained from the disk to normally run the APP.

[0043] In this implementation, to reduce the size of the installation package, the installation package includes the class description file instead of the class. Therefore, to ensure normal running of the APP, the class needs to be first generated based on the class description file.

5 **[0044]** For example, after downloading the installation package, the user can store the installation package in the disk of the terminal device. When the APP runs, the class description file in the installation package can be read from the disk to a memory, where the class is generated based on the class description file.

[0045] In some implementations, referring to FIG. 4, a procedure of generating the class based on the class description file may include the following steps.

[0046] S41. Obtain a class name, a parent class name, and an attribute based on the class description file.

[0047] For example, the class description file includes a class name and attribute description of a class waiting to be generated. For example, the class name
15 (className) is cashier (MERCHANTAPPOperator), and the attribute description includes a cashier ID (operatorId), a cashier name (operatorName), a shop name (shopName) of the cashier, etc. Therefore, the class name and the attribute description can be obtained from the class description file.

[0048] After the attribute description is obtained, an attribute can be created based
20 on the attribute description. For example, an attribute variable (ivar), an accessor (property), and a pointer (IMP) can be directly created based on the attribute description.

[0049] If the class description file includes a corresponding parent class name of the class name, the parent class name can be directly obtained from the class
25 description file. If the class description file does not include a corresponding parent class name of the class name, the parent class name can be created and returned based on a parent class name creation rule and the class name, to obtain the parent class name.

[0050] S42. Generate the class based on the class name, the parent class name,
30 and the attribute and by using a method provided by an iOS runtime technology.

[0051] The iOS runtime technology provides a plurality of methods for creating the class. For example, one method is objc_allocateClassPair(Class superclass, const char *name, size_extraBytes).

[0052] Note: The first parameter superclass is a class object of a parent class, the

second parameter name is a class name, and the third parameter is an additionally allocated byte, which is generally 0.

[0053] The parent class name and the class name are used in the parameters of the method. In addition, the attribute is used when the method is performed, to create the class based on the parent class name, the class name, and the attribute and by using the earlier described method.

[0054] In this implementation, the installation package includes the class description file, and the size of the class description file is smaller than that of the class. Therefore, compared with an existing method of including a class directly in an installation package, including the class description file in the installation package can reduce the size of the APP installation package and reduce resource overheads.

[0055] FIG. 5 is a schematic structural diagram of an APP running apparatus, according to an implementation of the present disclosure. The apparatus 50 includes a generation module 51, a download module 52, and a running module 53.

[0056] The generation module 51 is configured to: generate a class description file based on an original file of an APP, and place the class description file in an installation package, the class description file is a text file used for class description, and the size of the class description file is smaller than the size of a corresponding class.

[0057] For example, the original file includes a header file (.h file) and a source file (.m file).

[0058] The class description file is a text file and used to describe related information of a class. Classes are basic information in computer programming. The size of the class description file is smaller than the size of a corresponding class.

[0059] For example, a class name (className) is cashier (MERCHANTAPPOperator), and the cashier has class description (classDescription). For example, the class description includes attribute description, which includes a cashier ID (operatorId), a cashier name (operatorName), a shop name (shopName) of the cashier, etc.

[0060] Optionally, the generation module is configured to generate a class description file based on an original file of an APP, which includes: performing format conversion based on a predetermined script, and converting the original file of the APP into the class description file.

[0061] For example, in the iOS system, a pre-compiled script can be used to

convert the original file into the class description file (.plist file). Content needed in the class description file can be found in the original file, but the original file and the class description file have different formats. Therefore, the script can be pre-compiled to implement conversion between the original file and the class description file.

5 **[0062]** In addition, after the class description file is generated, it can be stored in a predetermined directory, for example, resource bundle.

[0063] After the class description file is generated, it can be placed in the installation package for a user to download and install.

10 **[0064]** In some implementations, referring to FIG. 6, the generation module 51 includes a first unit 511, a second unit 512, and a third unit 513.

[0065] The first unit 511 is configured to generate the class description file based on the header file and the source file.

[0066] For a generation procedure, refer to the earlier described related description. Details are not described here again.

15 **[0067]** The second unit 512 is configured to delete the source file.

[0068] The third unit 513 is configured to: place the header file and the class description file in the installation package for the user to download, after the source file is deleted.

20 **[0069]** Because the class description file and the header file are both small, and the obtained installation package is also small, the user can directly download the installation package from the APP store.

[0070] Optionally, the generation module 51 further includes: a fourth unit 514, configured to compress the installation package, so that the user downloads a compressed installation package.

25 **[0071]** To further reduce the size of the installation package, the obtained installation package can be further compressed.

[0072] For example, the compressed installation package is presented in the APP store, and the user can download the compressed installation package.

30 **[0073]** The download module 52 is configured to download the installation package before a user runs the APP.

[0074] For example, the user can download the installation package of the APP from the APP store, and the installation package includes the class description file.

[0075] Different from the existing technology, compiling and link packaging of the original file are not needed in this implementation, but the class description file is

used instead. The class description file is a text file. Compared with the existing ipa package, the size of the class description file is small, so as to reduce resources for download and storage.

[0076] The running module 53 is configured to: obtain the class description file from the installation package when the APP runs, and generate a class based on the class description file.

[0077] When the downloaded installation package is the compressed installation package, the installation package can be first decompressed when the installation packaged is used, so as to obtain information such as the class description file. When the downloaded installation package is not compressed, information such as the class description file can be directly obtained from the installation package.

[0078] To ensure normal running of the APP, the class included in the APP needs to be generated. In the existing technology, the class can be generated by compiling and packaging the source file, so that the ipa package stored in the disk includes the class. When the APP runs, the class can be directly obtained from the disk, to implement normal running of the APP.

[0079] In this implementation, to reduce the size of the installation package, the installation package includes the class description file instead of the class. Therefore, to ensure normal running of the APP, the class needs to be first generated based on the class description file.

[0080] For example, after downloading the installation package, the user can store the installation package in the disk of the terminal device. When the APP runs, the class description file in the installation package can be read from the disk to a memory, where the class is generated based on the class description file.

[0081] In some implementations, the running module 53 is configured to generate a class based on the class description file, which includes: obtaining a class name, a parent class name, and an attribute based on the class description file; and generating the class based on the class name, the parent class name, and the attribute and by using a method provided by an iOS runtime technology.

[0082] For example, the class description file includes a class name and attribute description of a class waiting to be generated. For example, the class name (className) is cashier (MERCHANTAPPOperator), and the attribute description includes a cashier ID (operatorId), a cashier name (operatorName), a shop name (shopName) of the cashier, etc. Therefore, the class name and the attribute description

can be obtained from the class description file.

[0083] After the attribute description is obtained, an attribute can be created based on the attribute description. For example, an attribute variable (ivar), an accessor (property), and a pointer (IMP) can be directly created based on the attribute description.

[0084] If the class description file includes a corresponding parent class name of the class name, the parent class name can be directly obtained from the class description file. If the class description file does not include a corresponding parent class name of the class name, the parent class name can be created and returned based on a parent class name creation rule and the class name, to obtain the parent class name.

[0085] The iOS runtime technology provides a plurality of methods for creating the class. For example, one method is objc_allocateClassPair(Class superclass, const char *name, size_t extraBytes).

[0086] Note: The first parameter superclass is a class object of a parent class, the second parameter name is a class name, and the third parameter is an additionally allocated byte, which is generally 0.

[0087] The parent class name and the class name are used in the parameters of the method. In addition, the attribute is used when the method is performed, to create the class based on the parent class name, the class name, and the attribute and by using the earlier described method.

[0088] In this implementation, the installation package includes the class description file, and the size of the class description file is smaller than that of the class. Therefore, compared with an existing method of including a class directly in an installation package, including the class description file in the installation package can reduce the size of the APP installation package and reduce resource overheads.

[0089] It is worthwhile to note that in the descriptions of the present disclosure, the terms "first" and "second" are merely intended for a purpose of description, and shall not be understood as an indication or implication of relative importance. In addition, in the descriptions of the present disclosure, "a plurality of" means "at least two" unless otherwise stated.

[0090] Descriptions about any process or method described in a flowchart or in another way here can be understood as follows: one or more modules, segments, or parts of code of an executable instruction that is used to implement a specific logical

function or operation step are included, the scope of preferred implementations of the present disclosure includes other implementation, and functions can be performed in a sequence other than a shown or discussed sequence, including in a basically simultaneous way or a reverse sequence according to the functions concerned. This
5 should be understood by a person skilled in the art of the implementations of the present disclosure.

[0091] It should be understood that parts in the present disclosure can be implemented using hardware, software, firmware, or a combination of them. In the earlier described implementations, a plurality of steps or methods can be implemented
10 using software or firmware that is stored in a memory and that is executed by an appropriate instruction execution system. For example, if hardware is used for implementation, being similar to another implementation, any one or a combination of the following well-known technologies in the art can be used for implementation: a discrete logic circuit having a logic gate circuit that is used to implement a logical
15 function for a data signal, an APP-specific integrated circuit having an appropriate combinatorial logic circuit, a programmable gate array (PGA), a field programmable gate array (FPGA), or the like.

[0092] A person of ordinary skill in the art can understand that all or some of the steps of the method implementations can be implemented by a computer implemented
20 program instructing relevant hardware. The program can be stored in a computer readable storage medium. When the program runs, one or a combination of the steps of the method implementations is performed.

[0093] In addition, functional units in the implementations of the present disclosure can be integrated into one processing module, or each of the units can exist
25 alone physically, or two or more units can be integrated into one module. The integrated module can be implemented in a form of hardware, or can be implemented in a form of a software functional module. When the integrated module is implemented in the form of a software functional module and sold or used as an independent product, the integrated module can be stored in a computer-readable
30 storage medium.

[0094] The earlier described storage medium can be a read-only memory, a magnetic disk or an optical disk.

[0095] In descriptions of the present specification, descriptions about such reference terms as "an implementation", "some implementations", "an example", "a

specific example", and "some examples" mean that specific features, structures, materials, or characteristics described with reference to the implementations or examples are included in at least one implementation or example of the present disclosure. In the present specification, the earlier described example expressions of
5 the terms are not necessarily with respect to a same implementation or example. In addition, the described specific features, structures, materials, or characteristics can be combined in a proper way in any one or more of the implementations or examples.

[0096] Although the implementations of the present disclosure are shown and described above, it can be understood that the earlier described implementations are
10 examples, and cannot be construed as a limitation on the present disclosure. Within the scope of the present disclosure, a person of ordinary skill in the art can make changes, modifications, replacements, and variations to the earlier described implementations.

CLAIMS

What is claimed is:

1. A method for running an application (APP), the method comprising:
generating a class description file based on an original file of an APP, wherein the class description file is a text file used for class description, and the class description file has a size that is smaller than a corresponding class size;
placing the class description file in an installation package (S11);
downloading the installation package before a user runs the APP (S12);
determining whether the APP runs;
in response to determining that the APP runs, obtaining the class description file from the installation package; and
generating a class based on the class description file (S13).
2. The method according to claim 1, wherein generating the class description file based on the original file of an APP comprises:
performing conversion of a format based on a predetermined script; and
converting the original file of the APP into the class description file.
3. The method according to claim 1, wherein generating the class based on the class description file comprises:
obtaining a class name, a parent class name, and an attribute based on the class description file (S41); and
generating the class based on the class name, the parent class name, and the attribute and by using a method provided by an iOS runtime technology (S42).
4. The method according to any one of claims 1 to 3, wherein the original file comprises a header file and a source file.
5. The method according to claim 4, wherein generating the class description file based on the original file of an APP comprises generating the class description file based on the header file and the source file (S111).

6. The method according to claim 5, wherein placing the class description file in an installation package comprises:

deleting the source file (S112); and

after the source file is deleted, placing the header file and the class description file in the installation package for the user to download (S113).

7. The method according to claim 6, further comprising, in response to placing the header file and the class description file in the installation package,

compressing the installation package to generate a compressed installation package for download (S114).

8. The method according to any one of claims 4 to 7, further comprising:

compiling the source file to generate an intermediate file (S31);

processing a link of the intermediate file to generate the installation package (S32);

downloading the installation package in a terminal device (S33); and

running the installation package on the terminal device (S34).

9. The method according to any one of claims 1 to 8, wherein the class description file comprises an attribute description.

10. An apparatus for running an application (APP), the apparatus comprising a plurality of modules configured to perform the method of any one of claims 1 to 9.