

ANIMATION GENERATION METHOD AND APPARATUS

[0001] The present application claims priority to Chinese Patent Application No. 201610440283.0, filed on June 17, 2016 and entitled "ANIMATION GENERATION METHOD AND APPARATUS", which is incorporated here by reference in its entirety.

TECHNICAL FIELD

[0002] The present application relates to the field of image processing technologies, and in particular, to an animation generation method and apparatus.

BACKGROUND

[0003] A canvas element is a hypertext markup language (HTML) element, and the canvas element does not have a drawing function itself, but provides an application programming interface (API) to support drawing operations of a script client.

[0004] In related technologies, timeline-based tween technologies are usually used to generate an animation. However, in such an implementation, a plurality of canvas elements can be generated during the generation of a complex animation with a plurality of objects. A client draws a whole canvas for each canvas element when playing the animation. Consequently, relatively high performance overheads can be produced to the device.

SUMMARY

[0005] In view of this, the present application provides an animation generation method and apparatus, to alleviate a problem that relatively high overheads are caused when a complex animation is generated by using related technologies.

[0006] The present application is implemented by using the following technical solutions.

[0007] An animation generation method is provided, and the method includes: obtaining a quantity of tweens after receiving an animation generation instruction; generating one or more timelines when the quantity of tweens is greater than or equal to 2, where each timeline corresponds to one or more tweens, and a quantity of timelines is less than the quantity of tweens; and generating an animation for the
5 corresponding tween based on the timeline.

[0008] An animation generation apparatus is provided, and the apparatus includes a quantity acquisition unit, configured to obtain a quantity of tweens after an animation generation instruction is received; a timeline creation unit, configured to
10 generate one or more timelines when the quantity of tweens is greater than or equal to 2, where each timeline corresponds to one or more tweens, and a quantity of timelines is less than the quantity of tweens; and an animation generation unit, configured to generate an animation for the corresponding tween based on the timeline.

[0009] It can be seen from the previous descriptions that, in the present application, during generation of the animation, the timelines less than the tweens are
15 generated. When playing the animation, user equipment does not need to draw canvases of the same quantity as the tweens, so that many unnecessary redrawing operations are reduced, and performance overheads of the user equipment caused when the user equipment plays a complex animation are reduced.

20 BRIEF DESCRIPTION OF DRAWINGS

[0010] FIG. 1 is a schematic flowchart illustrating an animation generation method, according to an example implementation of the present application;

[0011] FIG. 2 is a schematic diagram illustrating a key frame timeline, according to an example implementation of the present application;

25 [0012] FIG. 3 is a schematic flowchart illustrating a method for generating an animation for a corresponding tween based on a timeline, according to an example implementation of the present application;

[0013] FIG. 4 is a schematic structural diagram illustrating an animation generation device that an animation generation apparatus is applied to, according to
30 an example implementation of the present application; and

[0014] FIG. 5 is a structural block diagram illustrating an animation generation apparatus, according to an example implementation of the present application.

DESCRIPTION OF IMPLEMENTATIONS

[0015] Example implementations are described in detail here, and examples of the example implementations are presented in the accompanying drawings. When the following description relates to the accompanying drawings, unless specified
5 otherwise, same numbers in different accompanying drawings represent same or similar elements. Implementations described in the following example implementations do not represent all implementations consistent with the present application. Instead, they are only examples of apparatuses and methods consistent with some aspects of the present application that are described in detail in the
10 appended claims.

[0016] The terms used in the present application are merely for illustrating specific implementations, and are not intended to limit the present application. The terms "a" and "the" of singular forms used in the present application and the appended claims are also intended to include plural forms, unless otherwise specified in the
15 context clearly. It should be further understood that the term "and/or" used in the present specification indicates and includes any or all possible combinations of one or more associated listed items.

[0017] It should be understood that although terms "first", "second", "third", etc. may be used in the present application to describe various types of information, the
20 information is not limited to the terms. These terms are only used to differentiate information of a same type. For example, without departing from the scope of the present application, first information can also be referred to as second information, and similarly, the second information can be referred to as the first information. Depending on the context, for example, the word "if" used here can be explained as
25 "while", "when", or "in response to determining".

[0018] A tween, also referred to as a tween animation, is a gradient animation generated between two key frames (a start key frame and an end key frame). When generating a complex animation with a plurality of objects, an animation generation device usually generates a corresponding timeline for each tween based on tweens
30 that are set by a producer. When user equipment plays the animation, a canvas element is generated based on each timeline, and the user equipment draws a corresponding whole canvas for each canvas element. Consequently, many unnecessary redrawing operations are caused, and relatively high performance

overheads are brought to the user equipment.

[0019] To alleviate the previous problem, the present application provides an animation generation method, to effectively reduce performance overheads of a device when the device plays an animation.

5 **[0020]** FIG. 1 is a schematic flowchart illustrating an animation generation method, according to an example implementation of the present application.

[0021] Referring to FIG. 1, the animation generation method can be applied to an animation generation device. The animation generation device can include an intelligent end-user device such as a smartphone, a tablet computer, a personal digital
10 assistant (PDA), and a PC. The animation generation method can be understood as processing logic performed by a client that is loaded in the animation device and that has an animation generation function, and the animation generation method includes the following steps.

[0022] Step 101: Obtain a quantity of tweens after receiving an animation
15 generation instruction.

[0023] In the present implementation, to generate an animation, a producer can insert a selected picture into the client as a key frame, or a user can draw a key frame by using a drawing function provided by the client. Implementations are not limited in the present application. The key frame is equivalent to an original image in a
20 two-dimensional animation, and is a frame corresponding to a key action in movement or change of a role or an object.

[0024] In the present implementation, the producer can insert the key frame and set a tween based on a key frame timeline provided by the client. Referring to a key frame timeline shown in FIG. 2, scales on the key frame timeline are used to indicate
25 frame numbers of key frames. For example, scale 1 indicates a key frame with frame number 1, and scale 5 indicates a key frame with frame number 5. The producer can insert a key frame into the key frame timeline, for example, the producer can insert, at scale 1, the key frame with frame number 1, and insert, at scale 15, a key frame with frame number 15. After inserting the key frame, the user can select two key frames
30 used for a tween. Optionally, the user can select a frame number in the key frame timeline by using a mouse, and key frames on the two sides of the frame number are respectively a start key frame and an end key frame of the present tween. The client can generate the tween based on the start key frame and the end key frame.

[0025] For example, still referring to the key frame timeline shown in FIG. 2,

assume that the producer inserts three key frames, and frame numbers of the three key frames are respectively 1, 10, and 15. When the producer selects frame number 5, the client can determine that a start key frame of the present tween is a key frame with frame number 1, and an end key frame is a key frame with frame number 10.

5 **[0026]** In the present implementation, the producer can set a plurality of tweens based on needs and based on the key frame timeline provided by the client. For example, the producer can tap the key frame timeline twice to set two tweens. Referring to tween information shown in Table 1, a frame number of a tween's start key frame is 5, a frame number of the tween's end key frame is 8, a frame number of
10 the other tween's start key frame is 12, and a frame number of the other tween's end key frame is 18.

Table 1

	Frame number of a start key frame	Frame number of an end key frame
Tween 1	5	8
Tween 2	12	18

[0027] In the present implementation, the animation generation instruction is
15 usually triggered by the user after the user sets a tween, and the client can obtain the quantity of tweens after receiving the animation generation instruction.

[0028] Step 102: Generate one or more timelines when the quantity of tweens is greater than or equal to 2, where each timeline corresponds to one or more tweens, and a quantity of timelines is less than the quantity of tweens.

20 **[0029]** Based on step 101, after obtaining the quantity of tweens, the client can first determine whether the quantity of tweens is greater than or equal to 2. When the quantity of tweens is 1, the client can generate an animation with reference to an implementation solution in related technologies. When the quantity of tweens is greater than or equal to 2, the client can generate timelines less than the tweens, and
25 each timeline corresponds to one or more tweens.

[0030] In the present implementation, the quantity of generated timelines can be set by a developer. In an example, when the quantity of tweens is greater than or equal to 2, one timeline can be generated, and the timeline corresponds to all the tweens. In

another example, when the quantity of tweens is greater than or equal to 2, the quantity of the timelines generated can be a half of the quantity of tweens. For example, assume that the quantity of tweens is 8, four timelines can be generated, and each timeline corresponds to two tweens. Alternatively, assume that the quantity of
5 tweens is 9, four timelines can be generated, where three timelines each correspond to two tweens, and the remaining timeline corresponds to three tweens. Certainly, five timelines can be generated, where four timelines each correspond to two tweens, and the remaining timeline corresponds to one tween. When a plurality of timelines are generated, a tween corresponding to each timeline can be randomly determined or can
10 be determined based on a specific rule, for example, tweens are sequentially allocated to the timelines in ascending order of frame numbers of the tweens' start frames.

[0031] Step 103: Generate an animation for the corresponding tween based on the timeline.

[0032] Based on step 102, after the timeline is generated, the animation can be
15 generated for the corresponding tween based on each timeline.

[0033] In the present implementation, the generated animation is essentially Javascript code, and a browser of user equipment can run the Javascript code to generate an animation for the user to view. In the present step, because the quantity of generated timelines is less than the quantity of tweens that are set by the producer, a
20 quantity of timelines in the generated Javascript code is also less than the quantity of tweens. When running the Javascript code, the user equipment generates a canvas and draws based on the quantity of timelines in the Javascript code, thereby reducing many unnecessary redraws.

[0034] It can be seen from the previous descriptions that, in the present
25 application, during generation of the animation, the timelines less than the tweens are generated. When playing the animation, the user equipment does not need to draw canvases of the same quantity as the tweens, so that many unnecessary redrawing operations are reduced, and performance overheads of the user equipment caused when the user equipment plays a complex animation are reduced.

[0035] In an example, when a certain timeline is corresponding to a plurality of
30 tweens, referring to FIG. 3, and the generating an animation for the corresponding tween based on the timeline can include the following steps.

[0036] Step 301: Determine whether there is an M -level conflicting tween group in the plurality of tweens, where M is a natural number, and an initial value is a

default count initial value. If yes, step 302 can be performed.

[0037] In the present implementation, when a certain timeline corresponds to a plurality of tweens, a tween object of each tween can be first obtained, and a quantity of corresponding tweens is counted based on each tween object. When a quantity of
5 tweens corresponding to a certain tween object is greater than or equal to 2, in other words, two or more tweens in a plurality of tweens corresponding to the timeline have the same tween object, determine whether a tween time of any one of the two or more tweens conflicts with a tween time of another tween can be determined. If the tween time of the any tween conflicts with the tween time of the another tween, it can be
10 determined that there is at least one M -level conflicting tween group in the plurality of tweens corresponding to the timeline, and the M -level conflicting tween group includes said any tween and the another tween.

Table 2

Timeline	Tween	Tween object	Frame number of a start key frame	Frame number of an end key frame
Timeline A	Tween 1	Element A	5	8
	Tween 2	Element A	6	11
	Tween 3	Element B	2	7
	Tween 4	Element A	12	15
	Tween 5	Element B	8	10

15 [0038] For example, referring to the example in Table 2, assume that an initial value of M is 1, and timeline A corresponds to five tweens. In the present step, determine whether there is a level-1 conflicting tween group in the five tweens. A tween object of each tween can be first obtained, and then a quantity of tweens corresponding to each tween object can be counted. For example, tween object
20 element A corresponds to three tweens, and tween object element B corresponds to two tweens.

[0039] In the present implementation, it is determined whether a tween time of

any one of the three tweens corresponding to tween object element A conflicts with a tween time of another tween. Optionally, for tween 1, whether a tween time of tween 1 conflicts with a tween time of tween 2 can be determined, and whether the tween time of tween 1 conflicts with a tween time of tween 4 can be determined. The tween time can be represented by using an interval between a frame number of a tween's start key frame and a frame number of the tween's end key frame. For example, the tween time of tween 1 is [5, 8], and the tween time of tween 2 is [6, 11]. When determining whether the tween time of tween 1 conflicts with the tween time of tween 2, whether there is an intersection between the tween time of tween 1 and the tween time of tween 2 can be determined. There is an intersection between [5, 8] and [6, 11]. Then, it can be determined that the tween time of tween 1 conflicts with the tween time of tween 2 and that there is a level-1 conflicting tween group in tweens corresponding to tween object element A. The level-1 conflicting tween group includes tween 1 and tween 2. For ease of description, a tween in the level-1 conflicting tween group can be referred to as a level-1 conflicting tween.

[0040] In the present implementation, with reference to the previous solution, it can be further determined that the tween time of tween 1 does not conflict with the tween time of tween 4 and that the tween time of tween 2 does not conflict with the tween time of tween 4. Then it can be determined that there is one level-1 conflicting tween group in the three tweens corresponding to tween object element A, and the level-1 conflicting tween group includes level-1 conflicting tween 1 and level-1 conflicting tween 2.

[0041] In addition, for the two tweens corresponding to tween object element B: tween 3 and tween 5, it can be determined that a tween time of tween 3 does not conflict with a tween time of tween 5 and that there is no level-1 conflicting tween group in the two tweens corresponding to tween object element B.

[0042] In the present implementation, a tween that is not included in the level-1 conflicting tween group is a non-conflicting tween, for example, tween 3, tween 4, and tween 5 are non-conflicting tweens corresponding to timeline A.

[0043] Step 302: Update an M -level conflicting tween that corresponds to the latest end time in the M -level conflicting tween group with an $(M+1)$ -level conflicting tween, and determine whether there is an $(M+1)$ -level conflicting tween group in the $(M+1)$ -level conflicting tween, until there is no $(M+K)$ -level conflicting tween group in an $(M+K)$ -level conflicting tween.

[0044] Based on a determining result of step 301, when there is the M -level conflicting tween group in the plurality of tweens corresponding to the timeline, an M -level conflicting tween that corresponds to the latest end time in each M -level conflicting tween group can be updated with an $(M+1)$ -level conflicting tween. This is because a frame image that corresponds to the latest end key frame usually can best indicate what a producer wants to achieve. Therefore, when there is a conflicting tween group for the same tween object, a tween that corresponds to the latest end key frame is used. An M -level conflicting tween having the largest frame number of a tween end key frame in the M -level conflicting tween group can be updated with the $(M+1)$ -level conflicting tween.

[0045] The tween information shown in Table 2 is still used as an example. For the level-1 conflicting tween group, a level-1 conflicting tween having the largest frame number of a tween end key frame in the level-1 conflicting tween group can be updated with a level-2 conflicting tween. A frame number of an end key frame of tween 1 is 8, and a frame number of an end key frame of tween 2 is 11. Then, tween 2 can be updated with the level-2 conflicting tween. As such, timeline A corresponds to one level-2 conflicting tween: tween 2, and includes no level-2 conflicting tween group. Therefore, step 303 can be further performed.

[0046] Step 303: Based on the timeline, generate an animation for a corresponding non-conflicting tween and the $(M+K)$ -level tween.

[0047] In the present implementation, when there is no $(M+K)$ -level conflicting tween group in the $(M+K)$ -level conflicting tween, the animation can be generated for the corresponding non-conflicting tween and the $(M+K)$ -level tween based on the timeline. Table 2 is still used as an example. In the present step, an animation can be generated for tween 2, tween 3, tween 4, and tween 5 based on timeline A.

[0048] It can be seen from the previous descriptions that, based on the technical solution provided in the present application, when a timeline corresponds to a plurality of tweens, an animation can be generated for a tween that corresponds to the latest end time in conflicting tweens with the same tween object and conflicting tween time, so as to effectively avoid generating conflicting animations for the same tween object, and improve user experience.

[0049] In another example in the present application, based on the tween information shown in Table 2, assume that timeline A further corresponds to tween 6, a tween object of tween 6 is element A, a frame number of a start key frame is 10, and

a frame number of an end key frame is 12, a tween time of tween 6 is [10, 12]. In step 301, it can be determined that there are two level-1 conflicting tween groups in four tweens corresponding to tween object element A: tween 1, tween 2, tween 4, and tween 6. Level-1 conflicting tween group 1 includes tween 1 and tween 2, and level-1
5 conflicting tween group 2 includes tween 2 and tween 6.

[0050] In step 302, tween 2 that corresponds to the latest end time in level-1 conflicting tween group 1 can be updated with a level-2 conflicting tween, and tween 6 that corresponds to the latest end time in level-1 conflicting tween group 2 can be updated with a level-2 conflicting tween. Then, it can be determined, with reference to
10 an implementation method in step 301, whether there is still a conflicting tween group (which can be referred to as a level-2 conflicting tween group) in tween 2 and tween 6. Because the tween time of tween 2 is [6, 11], and there is an intersection between the tween time [6, 11] of tween 2 and the tween time [10, 12] of tween 6, it can be determined that there is a level-2 conflicting tween group, and the level-2 conflicting
15 tween group includes tween 2 and tween 6. As tween 6 corresponds to the latest end time, tween 6 can be updated with a level-3 conflicting tween. As such, there is no level-3 conflicting tween group in the four tweens corresponding to tween object element A, and an animation can continue to be generated. In step 303, an animation can be generated for tween 3, tween 4, tween 5, and tween 6 based on timeline A.
20 Details are omitted in the present application.

[0051] Corresponding to the implementation of the animation generation method, the present application further provides an implementation of an animation generation apparatus.

[0052] The implementation of the animation generation apparatus in the present
25 application can be applied to an animation generation device. The apparatus implementation can be implemented by software, hardware, or a combination of hardware and software. Software implementation is used as an example. As a logical device, the apparatus is formed by reading a corresponding computer program instruction in a non-volatile memory to a memory by a processor of the animation
30 generation device in which the apparatus is located. From the perspective of hardware, as shown in FIG. 4, FIG. 4 is a structural diagram illustrating hardware of an animation generation device in which an animation generation apparatus in the present application is located. In addition to a processor, a memory, a network interface, and a non-volatile memory shown in FIG. 4, the animation generation

device in which the apparatus in the present implementation is located can usually include other hardware based on actual functions of the animation generation device. Details are omitted here for simplicity.

[0053] FIG. 5 is a structural block diagram illustrating an animation generation apparatus, according to an example implementation of the present application.

[0054] Referring to FIG. 5, the animation generation apparatus 400 can be applied to the animation generation device shown in FIG. 4, and includes a quantity acquisition unit 401, a timeline creation unit 402, and an animation generation unit 403. The animation generation unit 403 can further include a conflict determining subunit 4031, an updating subunit 4032, and an animation generation subunit 4033.

[0055] The quantity acquisition unit 401 is configured to obtain a quantity of tweens after an animation generation instruction is received.

[0056] The timeline creation unit 402 is configured to generate one or more timelines when the quantity of tweens is greater than or equal to 2. Each timeline corresponds to one or more tweens, and a quantity of timelines is less than the quantity of tweens.

[0057] The animation generation unit 403 is configured to generate an animation for the corresponding tween based on the timeline.

[0058] The conflict determining subunit 4031 is configured to determine whether there is an M -level conflicting tween group in the plurality of tweens when the timeline corresponds to a plurality of tweens. M -level conflicting tweens in the M -level conflicting tween group have the same tween object and conflicting tween time, M is a natural number, and an initial value is a default count initial value.

[0059] The updating unit 4032 is configured to update an M -level conflicting tween that corresponds to the latest end time in the M -level conflicting tween group with an $(M+1)$ -level conflicting tween, and determine whether there is an $(M+1)$ -level conflicting tween group in the $(M+1)$ -level tween, until there is no $(M+K)$ -level conflicting tween group in an $(M+K)$ -level conflicting tween when there is the M -level conflicting tween group. K is a natural number greater than or equal to 0.

[0060] The animation generation subunit 4033 is configured to generate an animation for a corresponding non-conflicting tween and the $(M+K)$ -level tween based on the timeline.

[0061] Optionally, the conflict determining subunit 4031 is configured to obtain a tween object of each tween; when two or more tweens have the same tween object,

determine whether a tween time of any one of the two or more tweens conflicts with a tween time of another tween; and if the tween time of the any tween conflicts with the tween time of the another tween, determine that there is at least one *M*-level conflicting tween group in the plurality of tweens. The *M*-level conflicting tween
5 group includes the any tween and the another tween.

[0062] Optionally, the conflict determining subunit 4031 determines whether there is an intersection between a tween time between a start key frame and an end key frame of the any tween and a tween time between a start key frame and an end key frame of the another tween; and if there is an intersection, determines that the
10 tween time of the any tween conflicts with the tween time of the another tween.

[0063] Optionally, the updating subunit 4032 updates an *M*-level conflicting tween having the largest frame number of a tween end key frame in the *M*-level conflicting tween group with the (*M*+1)-level conflicting tween.

[0064] For an implementation process of functions and roles of each unit in the
15 apparatus, references can be made to an implementation process of corresponding steps in the previous method. Details are omitted here for simplicity.

[0065] Because an apparatus implementation basically corresponds to a method implementation, for related parts, references can be made to related descriptions in the method implementation. The previously described apparatus implementation is
20 merely an example. The units described as separate parts may or may not be physically separate, and parts displayed as units may or may not be physical units, and may be located in one position, or may be distributed on a plurality of network units. Some or all of the modules or units can be selected based on actual needs to achieve the objectives of the solutions in the present application. A person of ordinary skill in
25 the art can understand and implement the implementations of the present application without creative efforts.

[0066] The previous descriptions are merely example implementations of the present application, but are not intended to limit the present application. Any modification, equivalent replacement, or improvement made without departing from
30 the spirit and principle of the present application shall fall within the protection scope of the present application.

CLAIMS

1. A computer-implemented method, comprising:

receiving an animation generation instruction identifying key frames to use for generating an animation;

obtaining a plurality of tweens corresponding to the animation, wherein each of the tweens comprises a tween object between a start key frame and an end key frame;

generating one or more timelines when a quantity of tweens is greater than or equal to 2, wherein each timeline corresponds to one or more tweens, and a quantity of timelines is less than the quantity of tweens;

generating the animation using the plurality of tweens based on the one or more timelines, wherein generating the animation comprises:

when a given timeline of the one or more timelines corresponds to a plurality of tweens, determining whether there is an M -level conflicting tween group in the plurality of tweens, wherein M -level conflicting tweens in the M -level conflicting tween group have the same tween object and conflicting tween time, M is a natural number, and an initial value is a default count initial value;

when there is the M -level conflicting tween group, updating an M -level conflicting tween that corresponds to the latest end time in the M -level conflicting tween group with an $(M+1)$ -level conflicting tween, and determining whether there is an $(M+1)$ -level conflicting tween group in the $(M+1)$ -level tween, until there is no $(M+K)$ -level conflicting tween group in an $(M+K)$ -level conflicting tween, wherein K is a natural number greater than or equal to 0; and

generating an animation for a corresponding non-conflicting tween and the $(M+K)$ -level tween based on the timeline.

2. The computer-implemented method according to claim 1, wherein determining whether there is an M -level conflicting tween group in the plurality of tweens comprises:

obtaining a tween object of each tween; and

when two or more tweens have the same tween object, determining whether a tween time of any one of the two or more tweens conflicts with a tween time of another tween.

3. The computer-implemented method of claim 2, further comprising:

if the tween time of the any tween conflicts with the tween time of the another tween, determining that there is at least one *M*-level conflicting tween group in the plurality of tweens, wherein the *M*-level conflicting tween group comprises the any tween and the another tween.

4. The computer-implemented method according to claim 2, wherein the determining whether a tween time of any one of the two or more tweens conflicts with a tween time of another tween comprises:

determining whether there is an intersection between a tween time between a start key frame and an end key frame of the any tween and a tween time between a start key frame and an end key frame of the another tween, and if there is an intersection, determining that the tween time of the any tween conflicts with the tween time of the another tween.

5. The computer-implemented method according to claim 1, wherein updating the *M*-level conflicting tween that corresponds to the latest end time in the *M*-level conflicting tween group with an (*M*+1)-level conflicting tween comprises:

updating an *M*-level conflicting tween having the largest frame number of a tween end key frame in the *M*-level conflicting tween group with the (*M*+1)-level conflicting tween.

6. The computer-implemented method according to claim 1, wherein the *M*-level conflicting tweens in the *M*-level conflicting tween group is determined to have conflicting tween time based on an intersection between tween times between start key frames and end key frames of the conflicting tweens.

7. The computer-implemented method according to any one of claims 1 to 6, wherein generating the animation comprises:

using a single tween object from a plurality of tween objects having conflicting tween time.

8. An apparatus for generation of an animation, the apparatus comprising a plurality of modules configured to perform the method of any one of claims 1 to 7.